Polytechnic of Turin

Master's degree course in Mechatronic Engineering
A.y. 2022/2023
20th December 2022

Modelling of an electric induction motor on FPGA for testing in HIL environment

Presenters:
Alessandro Rizzo
Antonio Vitale

Candidate:
Fabio Oreiller

DEDICA

Prima di iniziare la lunga trattazione della tesi mi fa piacere spendere alcune parole per ricordare cosa conta davvero nella mia vita.
Alle porte di un traguardo così importante mi viene spontaneo guardarmi indietro e perdermi in un flusso di ricordi costellato da tante tappe; è una ferita ancora aperta quella dell'ultimo anno di liceo. Tutto sommato non è andata poi così tanto male alla fine, no?
Il percorso parte da una sconfitta, quella del liceo, e da una scommessa, quella di fare il test ingresso per ingegneria. E ora siamo qui, 5 anni dopo, a celebrare la fine di questo splendido percorso.
Spesso si sente dire che "l'università è il periodo più bello della vita". Non sono sicuro di questa cosa, ma sicuramente ricorderò per sempre con grande affetto ogni secondo che ho vissuto dentro le mura del politecnico: le persone, i posti e le emozioni che ho vissuto faranno parte di me e lo faranno per sempre. Da ognuno di questi momenti mi porto via un pezzo che ora è parte di me. Sono estremamente fortunato e grato di ciò che ho vissuto.
Questo percorso ha il mio nome e la mia faccia, ma non avrei mai potuto portarlo a termine se non avessi avuto accanto la mia mamma Mai'na gioia, il mio papà Ciccio, mia sorella Stuuupida e la mia ragazza PorGiGGiLek-Sube. La dolcezza e l'empatia di mamma, la trasparenza e la razionalità di papà, la tenacia e la costanza di Silvia, la spontaneità e l'allegria della mia Sube sono solo alcune delle costanti che mi hanno accompagnato in questi anni. Vorrei imparare da ognuno di voi. Io sono solo il frontman di questa splendida squadra che mi è sempre stata accanto e a cui devo tutto. A voi dedico tutto ciò di buono che mi è capitato in questo cammino, e se oggi sono fiero di quello che ho fatto e di ciò che sono, lo devo interamente a voi.
Questo percorso di tesi è la conclusione di un viaggio durato cinque splendidi anni, arricchito di tanti ricordi. Spero, con il tempo, di riuscire a rendervi anche solo un pezzettino di quello che ho ricevuto da tutti voi, e voglio provarci partendo da qui, da oggi. E questa è una promessa.
Non basterebbero pagine e pagine per dire tutto quello che vorrei e che sento: sono fortunato ed immensamente grato di quello che ho ricevuto, perciò grazie. Grazie di cuore e grazie per tutto.
Sono successe tante cose in questi ultimi anni, abbiamo tagliato tanti traguardi, e tante cose ancora succederanno, ma di una cosa sono certo: il meglio è davanti a noi.

Dedico tutto questo a voi.
Vi voglio bene.
Con affetto,
Fabio.

DEDICATION

Before I start my long thesis, I'd like to spend some words to remember what really matters in my life.

At the gates of such an important goal, I spontaneously look back and get lost in a flow of memories dotted with many stages; it is an open wound that last year of high school. It wasn't all that bad after all, was it?

The path starts from a defeat, that of high school, and from a bet, that of taking the entrance test for engineering. And now here we are, five years later, celebrating the end of this beautiful journey.

We often hear that "university is the most beautiful time of life". I am not sure about this, but I will surely remember forever with great affection every second I have lived within the walls of the Polytechnic: the people, places and emotions I have lived will be part of me and will do so forever. From each of these moments I take away a piece that is now part of me. I am extremely lucky and grateful for what I have experienced.

This path has my name and my face, but I could never have completed it if I had not had my mom Mai'na joy, my dad Ciccio, my sister Stuuupida and my girlfriend PorGiGGiLek-Sube. The sweetness and empathy of mom, the transparency and rationality of dad, the tenacity and perseverance of Silvia, the spontaneity and joy of my Sube are just some of the constants that have accompanied me in recent years. I would like to learn from each of you. I am just the frontman of this wonderful team that has always stood by me and to whom I owe everything. I dedicate to you all the good things that have happened to me on this journey, and if today I am proud of what I have done and what I am, I owe it entirely to you.

This thesis is the conclusion of a journey that lasted five splendid years, enriched with many memories. I hope, in time, to be able to make you even a little bit of what I have received from all of you, and I want to try starting from here, from today. And this is a promise.

Pages and pages would not be enough to say everything I want and feel: I am lucky and immensely grateful for what I received, so thank you. Thank you very much and thank you for everything.

Many things have happened in recent years, we have cut many goals, and many things will still happen, but I am sure of one thing: the best is yet to come.

I dedicate all this to you.
I love you.
With affection,
Fabio.

ABSTRACT

The automotive industry has always been one of the sectors that is constantly improving. Today, manufacturers' goals are moving towards the development of cars powered by electric motors. This choice is based on the need to reduce emissions in the environment. The presence of electric batteries has led to the development of new control units that communicate with the rest of the car. Automotive norms are very stringent and require a lot of testing in the field of safety.

From the union of these two themes arises the thesis in question, that is the configuration of a parametric electric motor for real time tests on HIL.

To have a first quick introduction to the contents of the thesis you can start from the analysis of the thesis title: Modelling of an electric induction motor on FPGA for testing in HIL environment. The electric motor chosen in the thesis is the induction motor, which bases its operating principle on the induction of a current, and therefore of a torque, in the rotor circuit by the stator.

The second focus of the title should be sought on the word FPGA: the tests that were performed in real time, but given the dynamics of the current the implementation of the project on the processor would not be enough to be able to correctly trace the variables. These are currents with dynamics of the order of ns. Hence the choice to move part of the project to FPGA, that is an electronic device characterized by large computing powers that allowed to handle these quantities.

Finally, we will talk about HIL, that is a frontier of the test-bench widely used in the automotive sector as it allows the test of a physical component with all its inputs/ outputs simulating the entire outline. In other words, the HIL allows you to test a low power component, such as a control unit, a relay, a butterfly valve, in its real working conditions simulating everything that is high power.

The conclusion of the thesis highlights the performance of the motor model with respect to speed targets, its strong dependence on the parameters that are passed and how the quantities evolve in real time by means of oscilloscopes linked to the HIL.

SUMMARY

LIST OF FIGURES

OBJECTIVE OF THE THESIS

The objective of the thesis in question was to configure a motor and an inverter for Hardware In the Loop (HIL) testing.
The modeled system serves as a testbench for external devices.
Speaking in detail of the thesis in question, the DUT for which the engine has been configured is a control unit. In real cases, therefore, an external control unit is connected to the HIL in order to see its correct functioning. Due to the costs of a real external control unit, in the project under examination, this was only simulated on FPGA. Nevertheless, the engine model has been configured in all its aspects as if there were an external ECU. In other words, if one day a real control unit arrives to be tested, it would be enough to connect it to the HIL and the whole system would allow the test. For now, for educational purposes, this has been simulated and instead of external wiring, the output signals from the simulator are directly sent back in and reworked.
During the following pages will be given first a general theoretical explanation of the components, then the model will be explained, paying attention only to some components (to see the others please, refers to *Luca Mutton* thesis), and the SW procedure. Only at the end will be reported the results obtained and the appropriate considerations will be made.

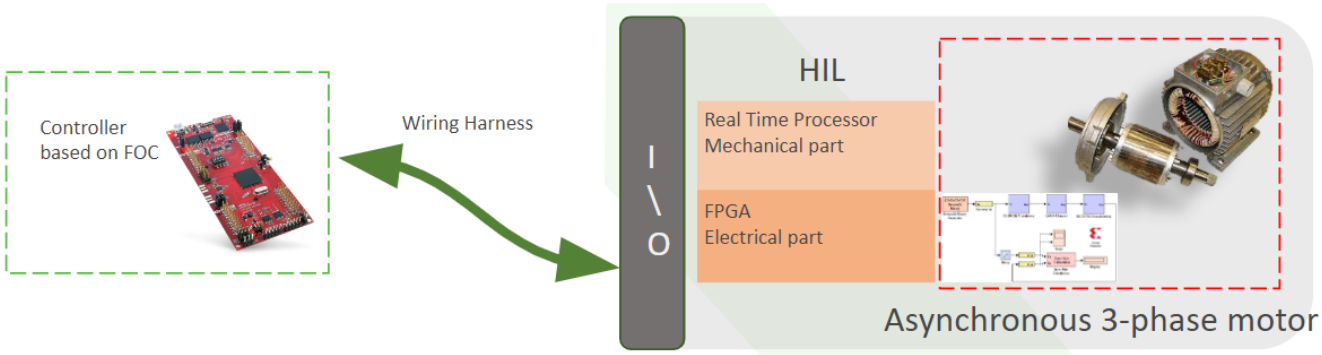


Figure 1 HIL-DUT interface

1. TOOLCHAIN

In order to achieve the goal just described, a rather rich tool chain was used, starting from a series of software tools up to the use of physical hardware.
Following the design order used during the work, the first software used has been MATLAB. In particular, extensive use of Simulink was made, which made it possible to model all the components of the thesis, both during the off-line design phases, therefore with a simulation on the processor, and during the real-time design phases.
Later in the thesis it will be explained why it was decided to switch to design using FPGAs. Therefore, it was necessary to use some Xilinx libraries. In particular, Xilinx for DSP has been useful to handle operations with digital quantities, such as the one produced by the FPGA. More than that, it has been used also System Generator, needed to specify the target hardware architecture.

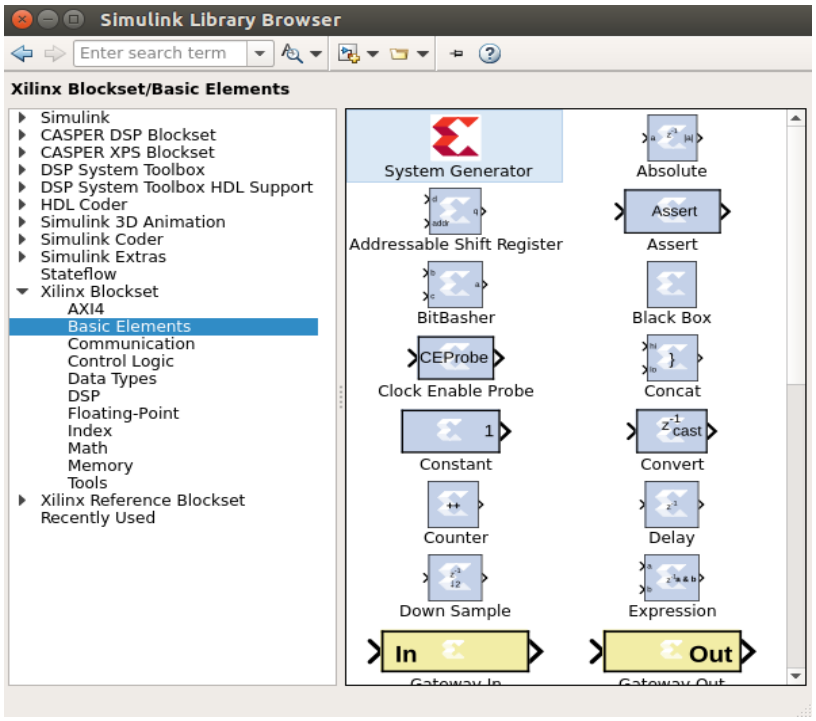


Figure 2 Xilinx library for FPGA (casper-toolflow.readthedocs)

Subsequently, to then move on to an online simulation, a tool to create the executable file to move the simulation to the HIL was needed. In particular, have been used all the blocks of the dSPACE RTI library. These blocks have been implemented to link together the FPGA one side with the physical channels, and on the other side with the processor.
Lastly, the Configuration desk and Control desk tools, both from dSPACE, were used. The first is necessary to create the executable file that can be put on the simulator: it is a program

necessary to configure all the physical input and output ports present on the In the Loop hardware.

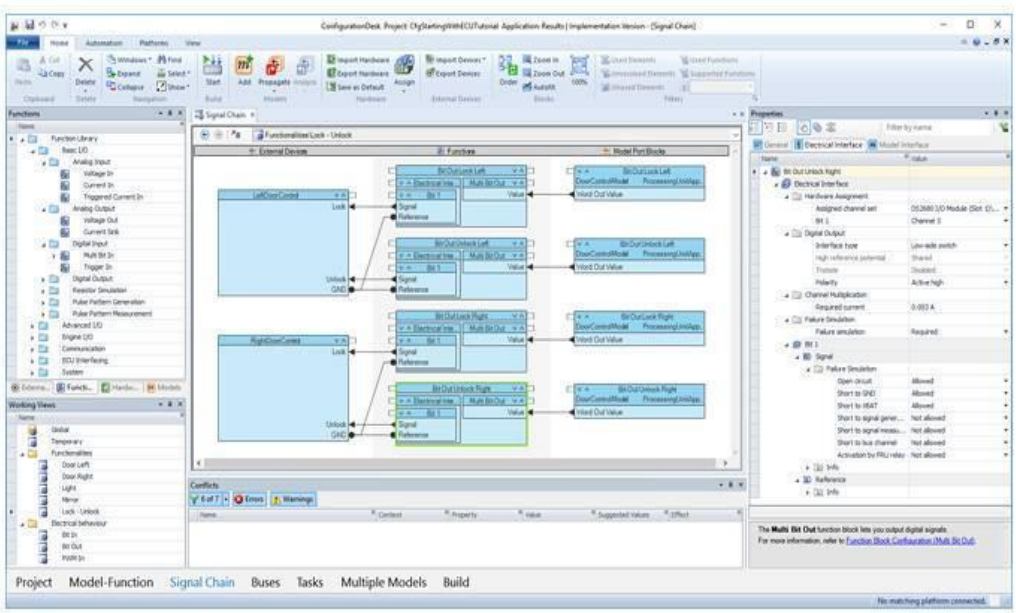


Figure 3 Configuration Desk layout (screen from Configuration Desk)

The second, on the other hand, is the application needed to run the simulations. This is an interface in which you can view all the quantities present within the model.

Figure 4 Control Desk layout (screen from Control Desk)

2. AUTOMOTIVE

The final result of the thesis is to configure an electric motor model for HIL testing of an external control unit.
Wanting to give a brief historical hint, the first electric car experiments date back to 1835, when the Dutch professor Sibrandus Straitingh began to think about a first model of electric car. Subsequently, the technological evolution leads in the 60s and 70s of the nineteenth century to another important stage, namely the realization by two French engineers, Gaston Planté and Camille Faure, of an electric propulsion car with such high performance as to be competitive with the steam engines of the time.
The potential of electric vehicles was clearly visible even at the time, so much so that the first vehicle capable of exceeding 100 km per hour was built in 1899, and it was an electric vehicle. The trend towards electric mobility continued to grow even during the first industrial revolution: at that time electric vehicles were able to reach a speed of 50 km / h, and therefore were ideal for urban circulation. To this advantage we must also add the fact that at the time they were easier to drive, required less maintenance and were much quieter than steam vehicles.
Subsequently, the development of electric technology suffered a sharp slowdown during the second industrial revolution, when a series of scientific innovations led to a real boom in petrol vehicles: first of all, special silencers were discovered for the noisy mufflers of the time, in secondly, radiators were introduced in order to solve all the problems of overheating typical of the internal combustion engine, and finally the starting through the electric motor was discovered, which simplified the awkward ignition with the crank. For all this series of reasons, cars with internal combustion engines began to be produced in series and the idea of the electric was abandoned for many years.

Today's electric and modern cars have numerous advantages over an internal combustion car: first of all, they have a much lower environmental impact as the operation of the engine does not depend on combustion, therefore no greenhouse gasses and carbon dioxide are produced. Speaking of costs, electric cars have lower maintenance costs, as they have fewer mechanical components and therefore less liquids to top up, consequently all costs related to servicing and maintenance are reduced. Added to this is the fact that a full electric tank costs about half compared to a full tank of petrol or diesel. The last advantages that we can identify are the reduction of noise pollution and some administrative advantages.
The other side of the coin basically has three drawbacks: the first is the purchase cost of electric cars, which is much higher than classic cars. Secondly, the autonomy of the batteries is currently very low compared to a full tank of fuel. Always linked to this problem, that is the fact that the duration of a full tank of electric cars depends on the type of recharge, but even

in the case of a high wattage direct current recharge, a recharge takes much longer than a full tank from the gas station. Finally, the biggest problem related to lithium batteries is the difficulty of disposing of them in an eco-sustainable way. This last point is very actual, and presents a contradiction: on the one hand, the reduction of pollution linked to the reduction of CO2 emissions, on the other hand the need to produce electricity related to the power plants that are powered by fossil coal, and the impossibility of disposing of batteries in an environmentally friendly manner.

The growing interest in electric mobility and the development of high-performance electric cars, combined with the versatility of HIL testing for modern control units were the reasons that led to the development of this thesis.

3. VERIFICATION AND VALIDATION

It has been repeated several times as the goal of the thesis is to configure an engine that can accept an external control unit to perform tests in a HIL environment.
The term testing is rather generic and includes two very important concepts, namely verification and validation. Wanting to investigate in more detail these two concepts, with software validation is meant to assess whether the software in question is in compliance with the intended uses, while software verification is the process by which it is verified that the software meets the required specifications. By putting the two concepts together, verification tends to check that the product is modeled correctly, validation instead ensures that the application has been implemented in accordance with the specifications for which it was designed. Ultimately the validation checks that the software complies with the security requirements and regulations in force, while validation aims to check that all the required features of the software are actually executable from the control unit without losing any of the tasks.
In general, when software is released, thousands of security and performance specifications are associated with it. This is why often the test phase is one of the most critical in the market of a new product.

The main tools to perform a software test are Software In the Loop (SIL), the Hardware In the Loop (HIL) and the Vehicle In the Loop (VIL). The last, obviously, is valid only in the automotive field.
SIL is a test system that provides for the verification of the software in a completely virtual environment. This leads to many advantages: first of all these tests are performed much faster than real tests. Moreover, they are much cheaper as they allow the creation of thousands of real scenarios without having the expenses that would involve physically testing everything. Another important advantage is related to the fact that, since the test is completely virtual, the simulation test can be performed faster than it would in reality, differently from the HIL, which is a real-time simulation.
On the other hand, the HIL development is based on testbenches that receive input from physical devices such as radar, cameras or external control units. In this scenario, the analyzed ECU responds during the test to external inputs as it would in its normal working conditions.
The latest frontier of the automotive test is the Vehicle In the Loop, which allows to fill the final gap between a simulation and a real road test that allows the validation of the complete vehicle.

Although these 3 types of verification are presented separately, they are usually performed in succession going in parallel with the development of the software. When the system is raw

it is easier to expect bugs and errors, which require thousands of tests to be detected and fixed. It would be too expensive to write every time different software versions, load them on a vehicle and run the tests. For this reason, SIL is mainly used in the first phase of the development of a vehicle. As the software becomes more robust, then you configure the HIL so you can test the components more and more at a high level. If in the SIL the functions are individually tested for bugs, in the HIL an integer component occurs going to see how it responds to real physical inputs. Once all the components are working, the VIL test is carried out, which is the most expensive due to the required equipment and the required computing power.

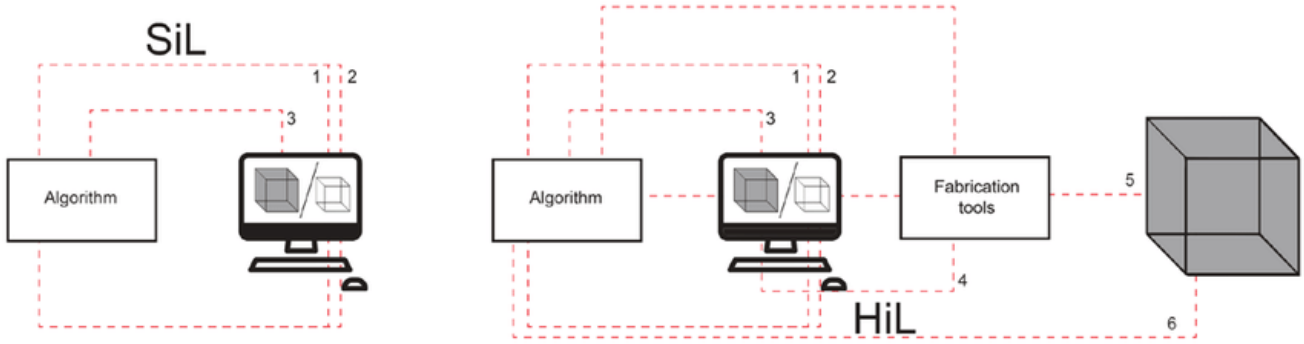


Figure 5 HIL and SIL (Software in the loop (SiL) and hardware in the loop (HiL) (Authors: L. Nyka and J. Cudzik))

WHAT IS AN HIL?

HIL simulation has become the standard for the development and validation of complex systems. Its development is linked to the search to find the right trade-off between the time of placing a product on the market and the complexity of the system. Classic tests involve downloading the software on the real system, which can be expensive and unsafe. The HIL test instead offers a valid alternative as the plant to test is replaced by an equivalent model able to be simulated in real time on a simulator specially configured, equipped with all cabling and real I/O capable of interfacing with an external ECU.
Ultimately this type of test is particularly efficient as it allows you to simulate the real dynamics of the plant.

Main Advantages

This type of test has become particularly important for a couple of reasons: first of all for a matter of safety if the components to be tested should be dangerous from the point of view of the safety of the tester. In general, a test in the car on the real component is much more effective (compared to a software test) to see the real behavior of the vehicle and the control

unit, but when scenarios involving the breakdown of some component or some electrical fails are to be tested, having the possibility to test the behavior of the control unit in a safe and controlled way can make the difference.
The second reason instead has to do with costs: during the design phase, the tests to be performed on a single component or a single function can be hundreds, so reducing costs is one of the first objectives in the research and development phases. Hence the usefulness of being able to simulate and test a component without the need to physically have it. This discourse takes on even more clarity when you think about all the tests that aim to validate the behavior in the event of failures or malfunctions. It is clear that physically breaking an engine or steering gearbox can be very onerous, especially if for many tests.

HIL components

To be more precise on the hardware components used, the most important is certainly the HIL Scalexio, from dSPACE. It is a modular simulator characterized by a high-performance processor, necessary for the request of real time calculations on electrical quantities and to manage external input and output peripherals.

Figure 6 HIL Scalexio (dspace.com)

The DS2680 card was then used. It is a compact input and output unit compatible with the Scalexio system, capable of providing numerous input and output channels necessary for simulation in the automotive field and for the connection with external control units. This card is able to receive both analog signals, such as voltages or currents, and digital quantities such as PWMs. The same quantities are also managed as output.
In the previous image can be seen that in the Scalexio it already mounted the DS2680

Figure 7 DS2680 IO card (dspace.com)

The FPGA is implemented through the DS2655M1 board, which has chosen for its ability to handle high resolution signals in applications where high processing speed is required. For all these characteristics, the board just mentioned is widely used in the automotive sector, hence the choice to implement it within the thesis in question.

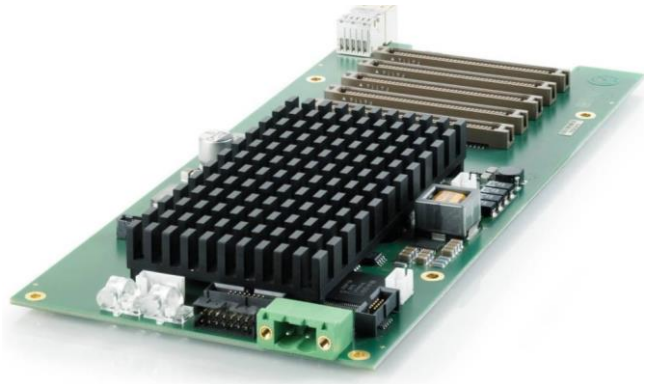


Figure 8 DS2655 M1 FPGA board (dspace.com)

4. ELECTRIC MOTORS

Having made this first general premise, now the attention of the discussion is shifted to the central theme of the thesis, namely the electric motor.
Today there are several electric motors on the market, used in almost every sector of the industry. Before going into the details of each of these, an electric motor is an electromagnetic component capable of converting electrical energy into mechanical energy. In general, it is an architecture composed of a fixed element, the stator, and a mobile one, the rotor. The latter is rotated by a magnetic field.
In general, the first criterion to choose the engine to be used must be to think about the type of movements required: some types of motors guarantee higher speeds, others allow greater dynamics in response to external elements, and so on. Then it becomes important to think about the final application: some engines are used in fields where traction is required rather than resistance to effort, other times it is preferable to have engines that require less maintenance.

Generally speaking, electric motors develop the maximum torque during acceleration from standstill. From a constructive point of view, this characteristic translates into the fact that electric motors do not need a starter motor. This, however, is only a consequence of the high torque: since they have great thrust almost with the engine stopped, then it is not necessary to use the classic hydraulic clutches. Finally, I am able to operate under multiple working conditions, thus not needing a change. This is also reflected in the possibility of reversing the polarity of the rotation with an electronic control. So reversing is superfluous. It is evident that electric vehicles have a much simpler kinematics that is reflected in yields of about 90%, much higher than the classic gasoline or diesel engines, which are around 30%.
Below is the characteristic Power-Torque of the electric motor to vary the speed of rotation. An important point of demarcation is the base speed: as you can see, in the area at lower speed of that base the motor behaves at constant torque, equal to the maximum value. After instead the voltage of the motor is maintained constant while the flow decreases, therefore the available torque decreases with hyperbolic course.
Power, on the other hand, has the opposite behavior: in the first part of the diagram the motor voltage increases constantly and so does the power, which grows in a linear way. After the base speed, the motor voltage reaches a value equal to the power supply, so the power stabilizes at a constant value.

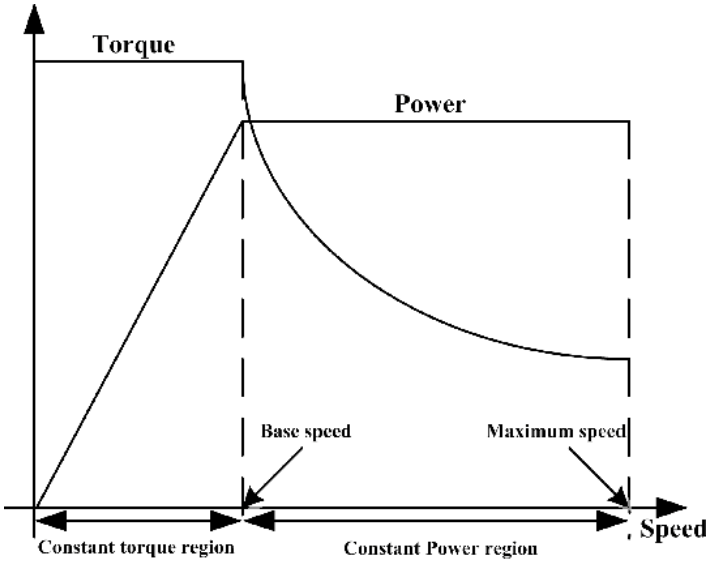


Figure 9 Torque-power characteristic (semanticscholar.org)

Electric motors are divided into four categories:

STEPPER MOTOR

To be precise, these motors are powered by direct current, but are generally used to divide a rotation into an equal number of steps. The motor under consideration is able to reach a precise angular position and maintain it without the need for feedback or sensors on the position by converting a series of input pulses (pwm) in an angular position of the rotor.
They are generally used in positioning control systems. Receiving discrete signals as input, they are controlled by digital signals and are implemented in open-loop systems.
Because of the characteristics described above, the industries that make the most use of step motors are those in which there is a need to position and maintain products in a precise position, such as in CNC machines, assembly, packaging and robotics.

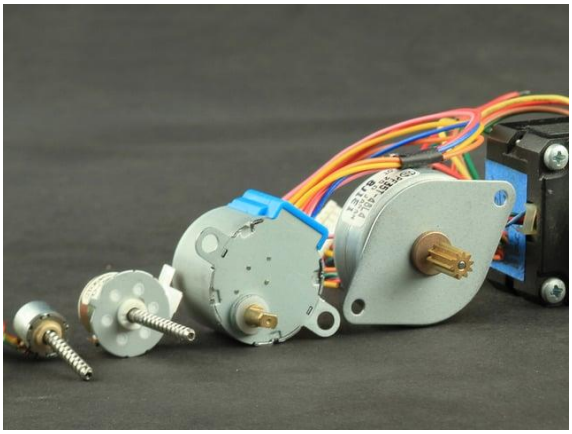


Figure 10 Stepper motor (learn.adafruit.com)

BRUSHLESS

Brushless motor falls into the category of DC motors with permanent magnets. As the name suggests, there is no contact with the rotor shaft, so the current circulates electronically. This generates a variation in the magnetic field that allows the rotation of the rotor. Since there are no contacts, the engine does not suffer slowdowns due to mechanical friction, so the performance is generally very high, standing around 98%. In other motors, brushes are the weak point, as they generate electromagnetic noise, like synchronous DC motors. The second advantage is that there are no sparks due to the brushes, which try to wear out the mechanical components. For this reason, the engine requires less periodic maintenance. Ultimately, brushless motors are able to generate great power compared to the footprint and ensure good accuracy on the rotor position, due to the low inertia of the rotor itself.
On the other hand, the biggest drawback is related to engine costs. If the brushed motors are controlled by a simple potentiometer on the brushes, the brushless are controlled using a controller. It is clear that the control carried out with a potentiometer is rawer than a microcontroller but extremely cheaper.
Because of the high performance are exploited in those applications where superconduction is required, such as the shipbuilding industry or those related to a high energy efficiency, where MW power is required. In recent years, however, this technology is also gaining ground in the automotive sector, in all contexts where fast and anticipated movements are required. To be able to guarantee such performances, as has been said, we need very small and sophisticated controllers. So, it's easy to imagine that cars made with this technology are very expensive. Mainly brushless motors are used in Formula E

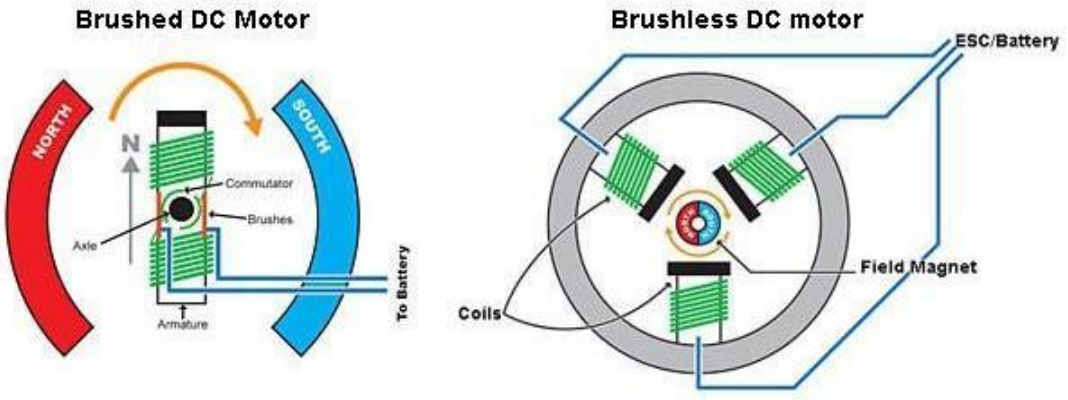


Figure 11 Comparison between brushed and brushless motor (haredataelectronics.co.uk)

DC MOTOR

From a constructive point of view, the DC motor is based on the use of a few components. Like the other models, the main players are the stator and the rotor. The stator generally consists of an electromagnet or a permanent magnet. The internal rotor windings are connected by a switch that serves to reverse the polarity of the magnetic field, allowing the machine to operate as a motor or dynamo (generator).
The great advantage of DC motors is linked to the ability to control with extreme precision the torque output and the number of revolutions of the motor. These motors are very versatile: they are used in small appliances but also in traction applications where large powers are required.
The biggest drawback is related to the friction with the brushes.

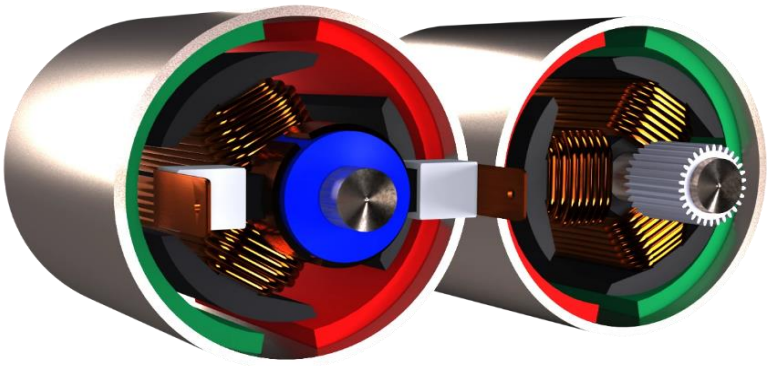


Figure 12 DC motor (oswos.com)

AC MOTOR

The AC motors are divided into two macro groups: synchronous and asynchronous:

Synchronous:

In the synchronous motor the stator contains magnetic coils connected to the motor control. The motion is due to the rotating magnetic field produced by the alternating currents flowing in the coils. This magnetic field drags the rotor which then produces a positive or negative velocity, depending on the orientation of the field itself in each polar pair. The higher the frequency of control, the greater the rotation of the motor. In addition, as the current supplied increases, the thrust provided by the electric motor increases.

The main advantage of this motor comes from the ease of controlling the rotor. Because it precisely follows the rotating field, controlling its rotation can impose a certain output speed. Another advantage is the particularly high performance as you should not "spend" anything to generate the rotating magnetic field.
On the other hand, however, to ensure a high magnetic field, the magnet must be built with special alloy materials, not the classic ferromagnetic materials. This definitely raises the costs of the synchronous motor. In addition, another drawback is related to the massive size of the motor because of the magnets.
Thanks to the high control precision, synchronous motors are generally implemented as an aid to thermal engines in hybrid vehicles.
A real car that mounts the synchronous electric motor is the Porsche Tycan.

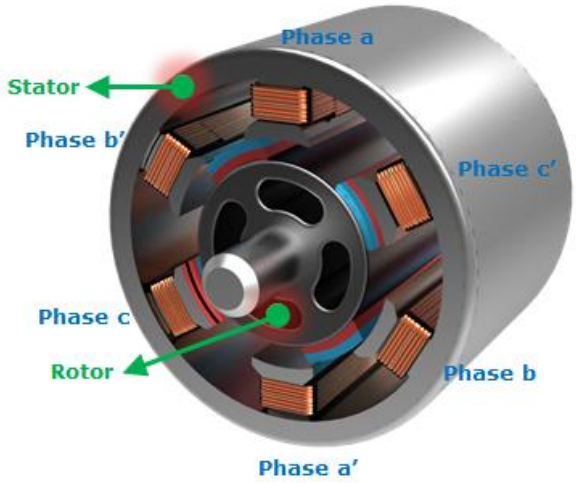


Figure 13 Synchronous motor (functionbay.com)

Asynchronous:

The AC motor was last held in this brief introduction as it deserves special attention, since it is the motor model of the thesis in question. To be precise, it is a squirrel cage induction motor, abbreviated SCIM. From a constructive point of view, rotor bars are inserted by casting and short circuited by using end rings. The following image shows the motor:

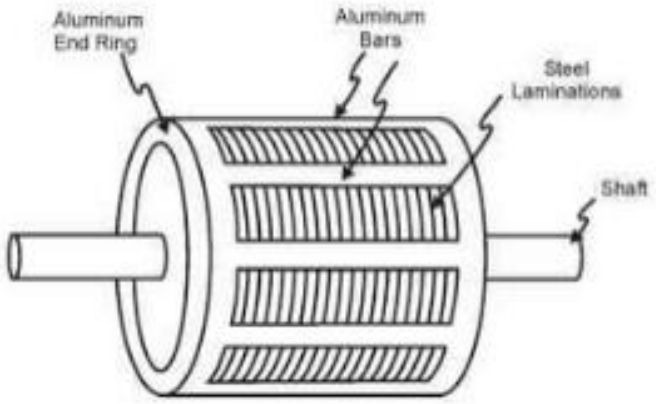


Figure 14 Squirrel Cage Induction Motor (Modern Electric, Hybrid Electric, and Fuel Cell Vehicles. (Author: Mehrdad Ehsani, Yimin Gao))

Then to better understand the working principle of the induction motor we also report a cross section to highlight some considerations:

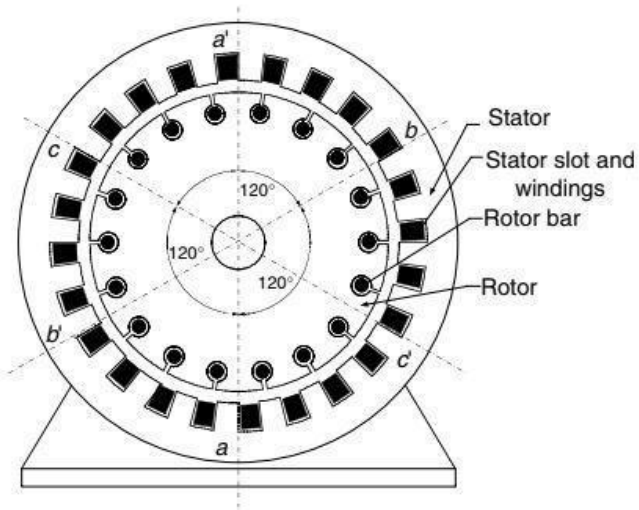


Figure 15 Cross section of a SCIM (Modern Electric, Hybrid Electric, and Fuel Cell Vehicles. (Author: Mehrdad Ehsani, Yimin Gao))

In the slots of the stator are inserted three phase windings a-a', b-b' and c-c'. The turns of each winding are distributed such that the current produces a sinusoidal flux density in the air gap.
The name of the induction motor itself implies that the torque is produced thanks to a force created by the rotor currents which are induced by a rotating field which is generated by the stator currents.
To be more precise we fed each phase with an AC sinusoidal current with a certain frequency ω and with a phase shift of 120^o one with respect to the others.

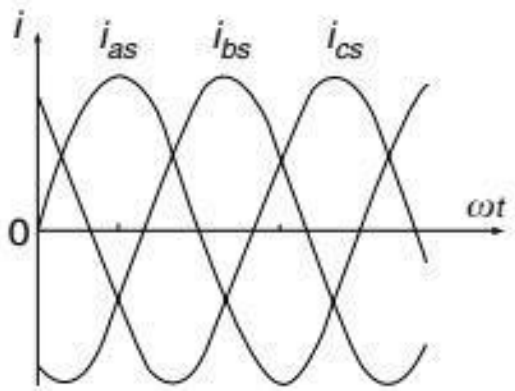


Figure 16 Three phase current (Modern Electric, Hybrid Electric, and Fuel Cell Vehicles. (Author: Mehrdad Ehsani, Yimin Gao))

The currents in the coils a-a', b-b' and c-c' produce alternate mmf F_{as}, F_{bs} and F_{cs}

$$F_{as} = F_{as}\ sin\ sin\ \omega t\ F_{bs} = F_{bs}\ sin\ sin\ (\omega t - 120^0)\ F_{cs} = F_{cs}\ sin\ sin\ (\omega t - 240^0)$$

The resultant stator mmf has the following expression

$$F_s = 3/2 F_s e^{(\omega t - 90^0)}$$

and can be immediately seen that the resultant mmf rotates with the same angular velocity ω of the three currents. Thanks to faraday's law, a rotating magnetic field induces a voltage and a current in the rotor conductors.

$$\varepsilon = -N \frac{\Delta\Phi}{\Delta t}$$

with:

- ε = induced voltage [V]
- $\Delta\Phi$ = rotating magnetic field [Wb]
- Δt = amount of time [s]
- N = Number of loops of the windings

According with Lorentz's force, if the rotor conductors are short circuited, then the induced currents generate a force on the rotor.

$$F = q(E + vxB)$$

with:

- F = Force [N]
- q = electric charge [C]
- E = external electric field $[\frac{N}{C}]$
- B = magnetic field [T]
- v = velocity $[\frac{m}{s}]$

This force is the source which generates the torque which moves the motor. Hence, we can notice that the induced current in the rotor is essential to produce a torque, and that the current depends on the relative movements between the stator mmf and the rotor.
This is why the motor is called asynchronous: because to have a current, and so a force and hence a torque, the rotor and the stator rotating magnetic field must have different velocity.

One of the main parameters of the asynchronous motor, which most differentiate it from the synchronous permanent magnet, and the slip parameter.

$$s = \frac{\omega_s - \omega_r}{\omega_s}$$

With:

- s = slip
- ω_s = stator speed [rpm]
- ω_r = rotor speed [rpm]

Precisely because of this great importance, the typical curve of the engine torque and reported to the variation of the slip.
In the study of the characteristic in question, the diagram can be subdivided in different working areas:

- $0 < s < s_{rated}$
  This first zone is characterized by the increase in torque proportional to the increase in the slip parameter. This is valid until the slip reaches the nominal value.

- s > s_{rated}
  The second area is considered unstable, in fact any type of variation leads to the machine stopping. This behavior is reflected in a torque that decreases sharply as the slip increases. This is not the only disadvantage of this part of the diagram, in fact high slip values also correspond to high currents that could damage the stator windings.

This division is based on the mathematical value of the slip, but the diagram can also be divided in another way to underline different working areas corresponding to an overall behavior of the motor:

- s < 0
  The first is where the slip is less than zero. if we reason on the slip formula, we can guess that if the slip is less than zero it means that the rotor speed is greater than that of the stator, therefore when the engine works in this area is producing a negative torque. This in the automotive world translates into reverse.
- 0 < s < 1
  The second working area is the one that provides a slip between zero and one. This is the classic condition in which the rotor pursues the stator at a lower speed. When the engine operates in such conditions a positive torque is produced, That translates into the forward march.
- s > 1
  When the creep is greater than one, the motor is acting as a generator and while producing positive torque.

Here it is reported exactly the same curve of before but seen in another way to underline the most important point of the curve:

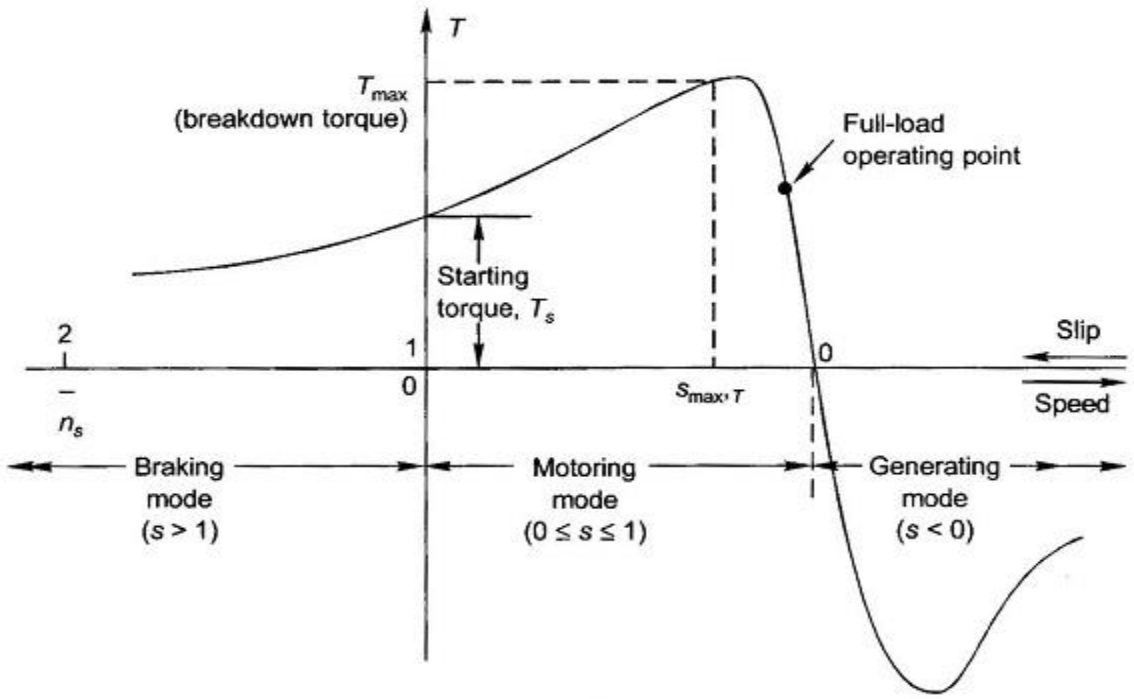


Figure 17 Key point of the Torque-speed characteristic (Modern Electric, Hybrid Electric, and Fuel Cell Vehicles. (Author: Mehrdad Ehsani, Yimin Gao))

- Starting torque is so called because it corresponds to s=1, which means that the rotor is still fixed. This situation occurs at the switch-on of the motor
- full load torque is the best operating point for the motor
- the maximum torque is called pull-out

Observing the trend of the feature just described, it is easy to guess the need for a way to control the speed of the engine, as the only favorable working area in the automotive sector and one that foresees a shift greater than zero but less than the nominal one, and this results in a small working area if compared to the whole feature.
Returning to talk about the asynchronous motor at a high level, the main advantage they offer is to be able to withstand an overload up to 2.5 times the rated power. This means that for a short time they can work beyond their optimal conditions. This possibility is limited by the risk of overheating of the coils. The engine therefore slows down a lot but does not change thanks to its robustness. The second advantage is related to the size: it is a lighter and more compact engine than the synchronous mentioned above.
However, it has a much lower yield, especially in the start-up phases. In order to operate, there must be a speed difference between the rotating magnetic fields of the stator and rotor, so the latter is delayed and less precise to control. For this reason, synchronous engines are generally preferred in the automotive world.
A real car that mounts the synchronous electric motor is the Tesla ModelS.

5. INVERTER

For now, we have talked about the electric motor model meaning only the motor itself, but this definition is incomplete. As we have just explained, the asynchronous motor works with alternating voltages despite the power battery of the electric motor is in direct current. To continue the discussion of the thesis is therefore necessary to talk about the inverter.
The inverter is an electronic device that can transform a direct current input in an alternating output changing the magnitude at the desired frequency. In general, there are many users that require alternating current to operate, so the use of the inverter is widespread in many sectors, from photovoltaic panels, to aerospace, in which they provide the alternating current necessary for avionics systems, up to the automotive sector. The latter case is the one analyzed in the thesis in question.
From a physic point of view, an inverter consists of a series of pairs of switches placed in parallel, which opening and closing generate a sinusoidal output while receiving as input alternate quantities.

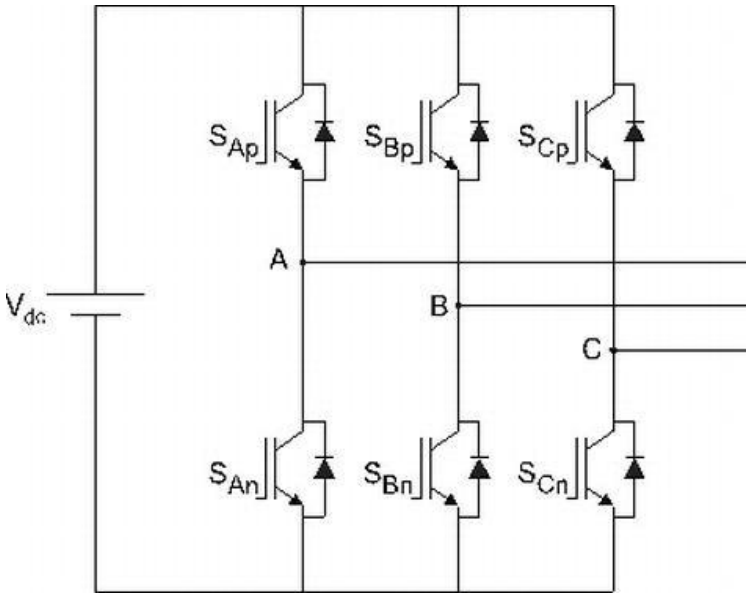


Figure 18 Electrical circuit of the inverter (Modern Electric, Hybrid Electric, and Fuel Cell Vehicles. (Author: Mehrdad Ehsani, Yimin Gao))

The signal cleaning changes according to the complexity of the inverter: the simplest ones consist of 4 switches that generate a square output voltage characterized by a module smaller than the input. Typically, downstream of these inverters is also put a transformer so as to provide the output of the required voltage. Due to the simplicity of the architecture, simpler inverters are generally replaced by simple MOSFET or IGBT transistors. Generally, devices powered by such inverters are characterized by electromagnetic noise.

The most modern and efficient inverters base their operating principle on a technique called pulse width modulation (PWM). The technician foresees the constant comparison between a reference sinusoidal signal with a triangular wave characterized by a much higher frequency. The higher the frequency, the better the inverter works. The comparator output is a binary signal, 1 or 0, based on the reciprocal magnitude of the triangular wave and the square wave.

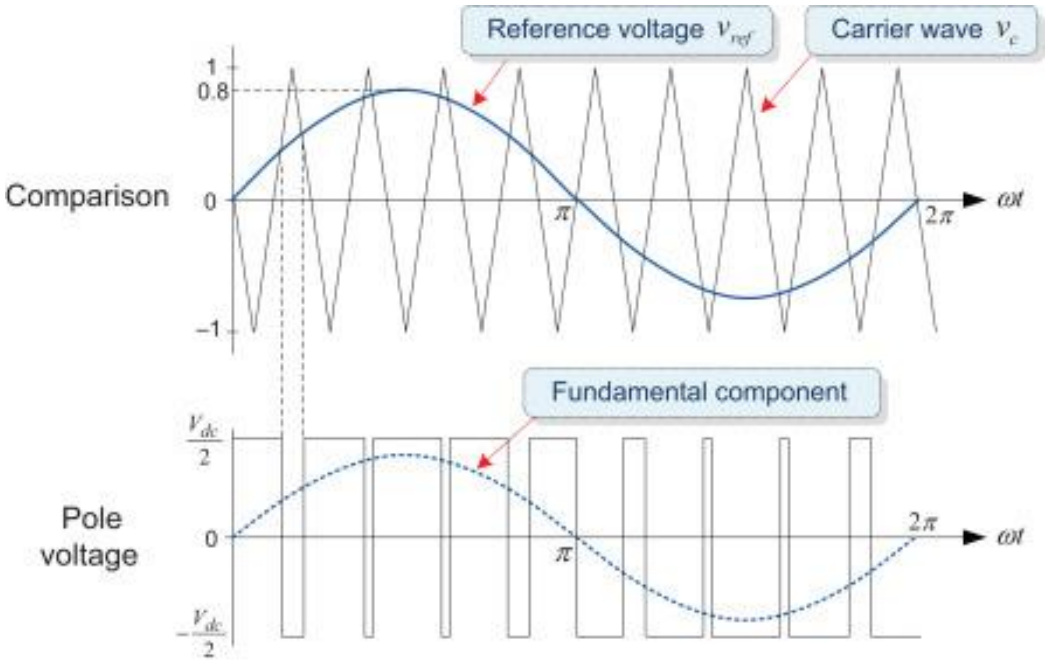


Figure 19 PWM modulation (sciencedirect.com)

The result of the comparison therefore generates a square wave that is sent to a MOSFET (or three pairs of switches in parallel) that generates a sine wave. This raw signal is then filtered by RC or RL circuits to obtain a cleaner signal.

Figure 20 Inverter (lifewire.com)

6. CONTROLLER

The control process is the key to many industrial processes since it mitigates unpredictable behaviors or errors due to external disturbances. A controller is fundamentals to refine the output to the desired one. In a very global way, process controllers work to converge the system to a certain set point: an input signal expressing the actual state of the system is sent to the controller. The controller compares its input to the set point and computes ad delta expressing the difference between these two signals. This delta is an error. Then controller makes the adjustments needed to move the operating point of the system such that it could reach the set point.
The latter took on even more importance when we switched from DC motor to AC motor. The first one, has always been exploited due to the ease of speed control, but they had numerous limitations, such as the need for frequent maintenance to the manifold, the limited rotation speed and little resistance to overload. For all these reasons, the transition to the AC motor was a direct consequence. These new motors are able to guarantee many advantages, at the expense of a more difficult control.
For these reasons, the controller and the control strategy have been important in the current project too.

The following image gives an idea of how many ECUs are present in a modern car. From the name it can be imagined how many aspects of the car are controlled by an ECU.

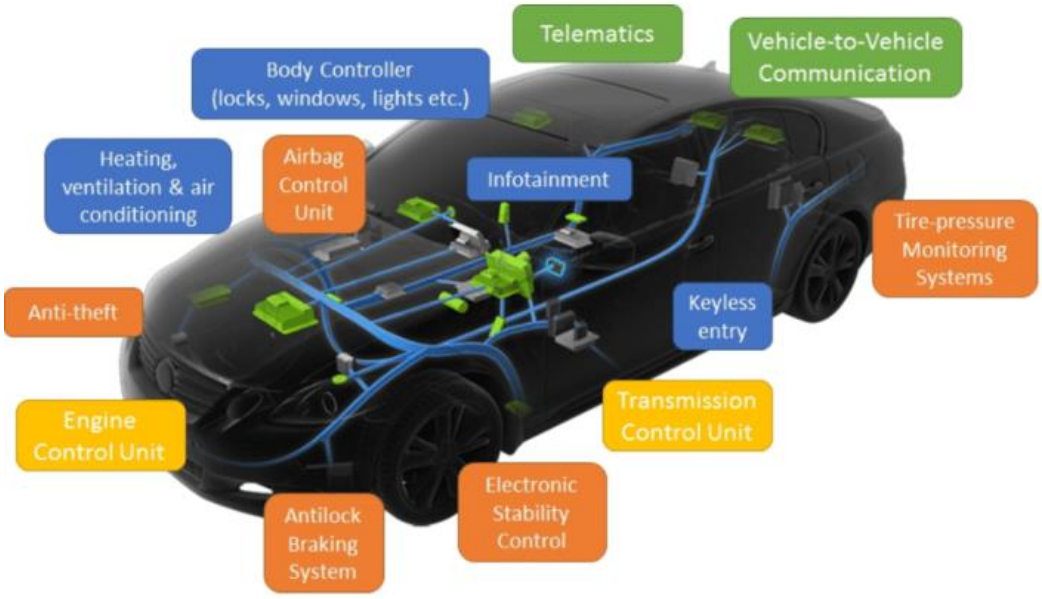


Figure 21 Different ECU present inside a modern car (Security Concerns in Co-operative Intelligent Transportation Systems (Author: Konstantinos Fysarakis, Vasilios Katos))

To be honest, the goal of the thesis should have been to configure a motor on the HIL, and so being able to accept an external device to test, but since the costs of a real control unit we've just simulated its behavior.
Below is a generic system in which you can distinguish the plant and the controller. In particular here is a closed chain system, but the distinction between plant and controller does not depend on the chain. The controller, as has been repeated several times, is what regulates the signals and quantities supplied to the plant, which instead is the set of the power converter and the load that undergo the corrective action.

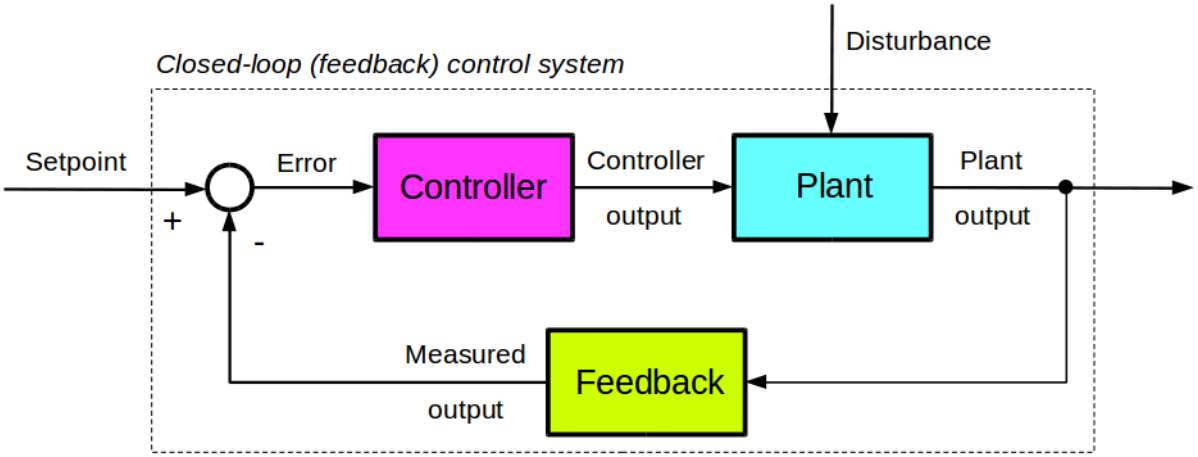


Figure 22 General overview of a system (x-engineer.org)

Talking about a controller the first big point is to distinguish between open loop and closed loop depending on the control strategy. The plant and the controller have different transfer functions in the two situations.

OPEN LOOP

In the open loop, the control action is independent from the output of the system. This means that the ecu doesn't care about what is going on in the system. This control strategy is used in not safety-critical applications. An example may be the following one: this scheme explains how a boiler works (the Heating Element). The input of the controller is the electrical energy, for sure, and a timer that can be set so that the boiler heated for a certain amount of time. Can be clearly seen that we can't have any certainty that the temperature in the room is what we want since there are no feedback of what is going on and the controller can't consider the presence of external disturbances.
Below the transfer function and a scheme are reported:

$$H_{ol} = H_c H_{plant}$$

Where:

- H_{ol} = transfer function of the open loop
- H_c = transfer function of the controller
- H_{plant} = transfer function of the plant

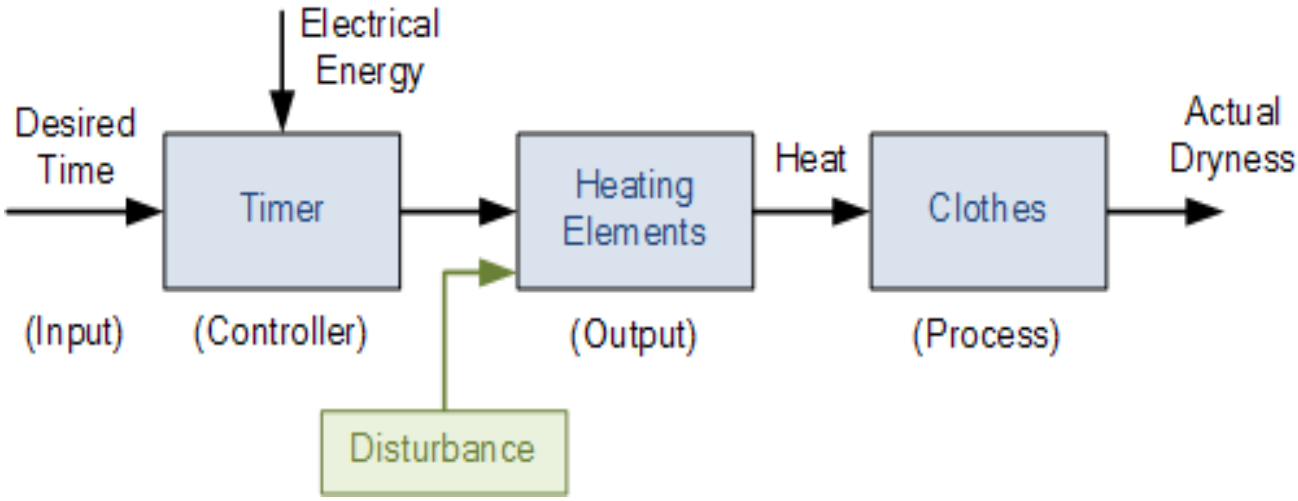


Figure 23 Open loop system (techgoggler.com)

CLOSED LOOP

A completely different working principle is typical of a closed loop controller, in which is kept in consideration the output of the system. To use the same situation of the open loop, to close the ring, in the system should be present a thermostat to monitor the building temperature, and thereby feedback a signal to ensure the controller maintains the building at the temperature set.

$$H_{cl} = \frac{H_{ol}}{1 + H_{ol}}$$

Where:

- H_{cl} = transfer function of the closed loop
- H_{ol} = transfer function of the closed loop
- H_{plant} = transfer function of the plant

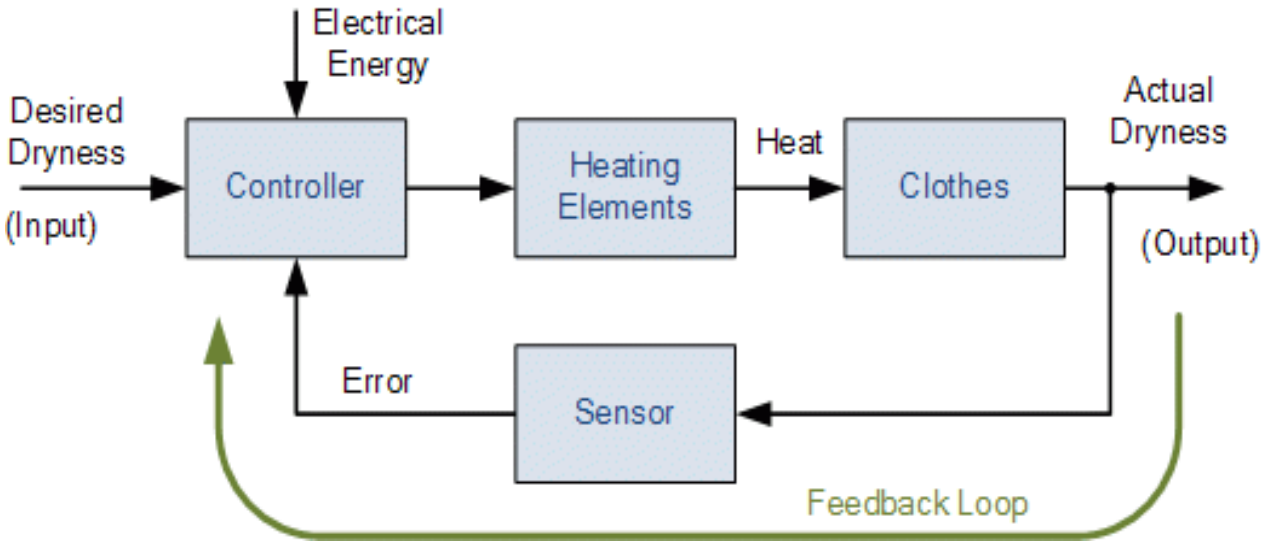


Figure 24 Close loop system (techgoggler.com)

The sensor measures the actual temperature and compares it with the desired one. Depending on the difference of them, applies a correction so that the temperature in the room is kept constant and equals to the set one.

Apart from the boiler, talking more in depth of the application, the controller works with a closed loop algorithm. This means that we can give a velocity as input of the system, and the controller try to generate a current able to control the motor such that it reaches the desired speed. Later on, it will be explained how is it possible.
The controller has many important tasks: the first is the one just explained, and so is needed by the motor to reach a target speed, while the second is to guarantee the correct torque for the final application of the engine.
In fact, regardless of the type of chain, the following is how the behavior of a plant under examination should be under the corrective action of a controller.

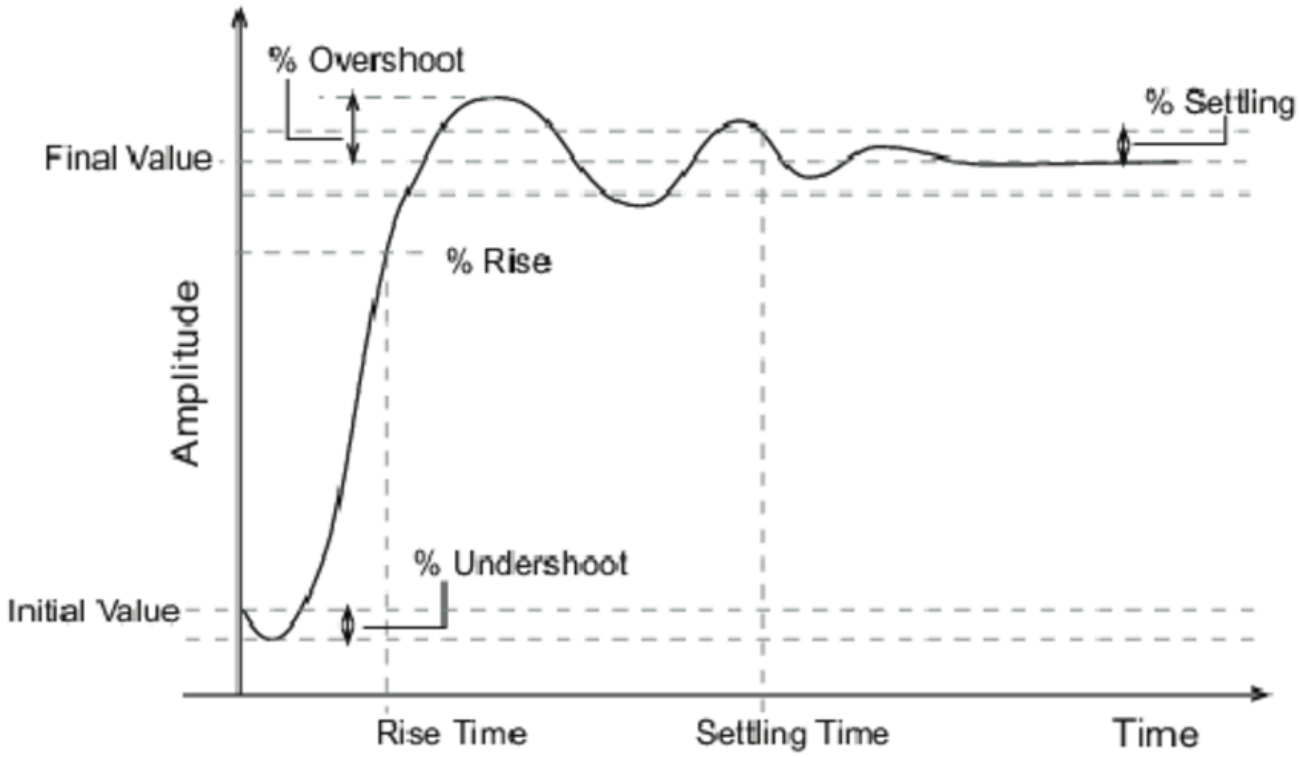


Figure 25 Steady state response (site.iugaza.edu.ps)

This diagram may be useful to explain what a controller should do: the black line expresses the dynamic behavior of the system. The goal of the controller is to impose the convergence of the actual speed (black line) as fast as possible to the final value, which is the target speed. As we can see, at the steady state the two lines are superimposed.
Just to link this graph with the previous argument, in general, as faster is the convergence to the desired value, as higher is the overshoot.

- Rise time: is the time needed from goes to the 10% to the 90% of the desired target. This parameter is a good example of how fast the response of the system is.
- Overshoot: it is the highest value reached by the output variable. As faster is the response as higher is the overshoot. In general, high peaks are unwanted in the final application. Thinking on our thesis situation, if we imagine of being in a car, we would like that it reaches the desired value in a smoothly, without jerking. This is way in general is not easy to correctly tune parameter of the controller. In general, the constants have opposite tendencies.

$$\hat{y} = \frac{\omega_{fdbk} - \omega_{ref}}{\omega_{ref}}$$

- Settling time: is the time needed by the controlled variable to reach the steady state value and remain inside an error band.
- Undershoot: deepest point of the system response to an external input

To conclude this first general analysis of a control system an ECU's image is reported.

Figure 26 Modern controller (ecu-tuning.com)

7.CLARKE & PARK TRANSFORM

In the discussion above it has been reported that the most effective control method for asynchronous motors is the vector one, that is the FOC. For these reasons the most obvious choice to model the controller of the thesis in question fell on a control FOC.
This algorithm calculates the currents needed to control the motor in a rotating reference system. This system is called d-q, or direct and in quadrature. These two references represent the decomposition into two of the rotating magnetic fields. The two components are responsible for the flow (d axis) and the torque (q) respectively.
The big advantage that comes from this transformation is that it allows to work on sinusoidal quantities (such as the analog ones of an electric motor) treating them as constant quantities. Hence the great accuracy of the FOC control.
In order to better follow the speech below, an image is attached on how the transformation takes place, followed by the mathematical equations that allow the transformation.

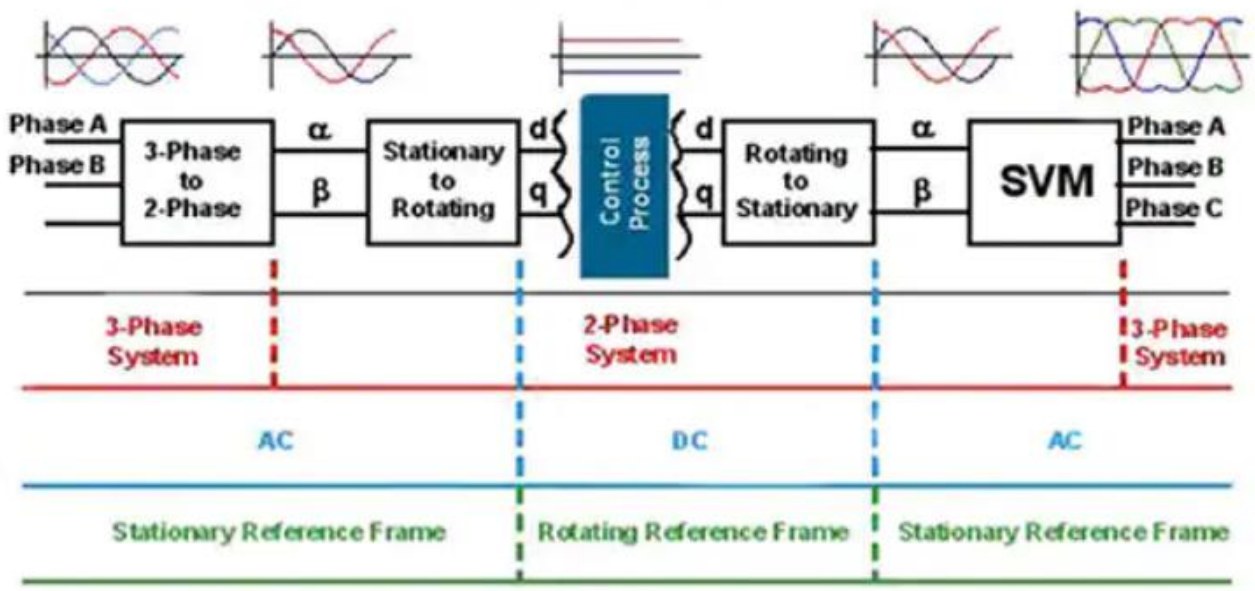


Figure 27 Clarke & Park transformation direct and inverse (ieeexplore.ieee.org)

This image is very clear. It takes into consideration almost entirely the project that is being treated. On the left is a signal in AC which is what comes out of the motor. Intuitively it is a three-phase alternating current.
The input magnitude undergoes a first partial transformation, from the three-phase time domain this is transformed into two components in a stationary orthogonal space (α, β).
This first operation is called Clarke transform.

$$[i_\alpha\ i_\beta\ i_0\ ] = \left(\frac{2}{3}\right)\begin{bmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} \\ 0 & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} \\ \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \end{bmatrix}(i_a\ i_b\ i_c\ )$$

But since the three currents must be balanced, which means that

$$i_a + i_b + i_c\ = 0$$

The previous matrix can be rewritten as follows:

$$[i_\alpha\ i_\beta\ ] = \begin{bmatrix} 1 & 0 \\ \frac{1}{\sqrt{3}} & \frac{2}{\sqrt{3}} \end{bmatrix}[i_a\ i_b\ ]$$

Then there is a transformation into a rotating orthogonal system. (d, q). This is the so called, Park transforms

$$[i_d\ i_q\ ] = [cos\theta\ sin\theta\ -sin\theta\ cos\theta\ ] * [i_\beta\ i_\beta\ ]$$

Returning to the high-level scheme, the two transformed in succession take as input a sinusoidal size and return as output two constant values that are sent to the controller. From here you can see how easy it is to implement precise control: the control unit has to manage constant quantities, which are much easier to treat than AC sizes. Once the control is made, the output passes through the anti-transformer Clarke & Park and then be provided to the inverter that runs the SVM (space width modulation) to control the motor the correct voltages in order to reach the target speed.

8. MODELLING PROCEDURE

After this long introductory overview, it is now time to go into detail on how the project was carried out in practice, by reviewing all the progressive versions of the model that have allowed the achievement of the final configuration and therefore the collection of results.
Just to be clear, all the models that are presented here shortly are part of the same project: it is not therefore experiments on their own but only steps that have led to the final result. This kind of engineering approach was chosen in order to better debug the system in case of errors: it would not have made sense to go directly to the complete final model, because in case of error it would not have been possible to identify the origin. In the end, the successive version was carried out only once we had the certainty of the correct functioning of the previous one, so that we could easily identify the problem in case of failure. Secondly, since the thesis involves a lot of effort in the use of new software (Configuration desk and Control desk), it was useful to proceed step by step.
At a high level the operating principle involves an electric motor powered by a battery that provides a constant voltage to the inverter. This has two inputs, the battery voltage and the generated PWM. The output of the inverter is a three-phase sinusoidal voltage that serves to power the motor, which generates a speed in rpm. The electrical angle of the rotating magnetic field of the rotor can be calculated from the output speed. This angle is fed to the blocks that calculate the direct transforms of Clarke & Park, necessary to generate a current in the reference frame d-q. In the IP, this current is compared with a reference value. The controller then reverses the Clarke & Park transforms so that the voltages return to a three-phase system to generate PWM.
Before a global overview it's reported to have an idea of the whole system.

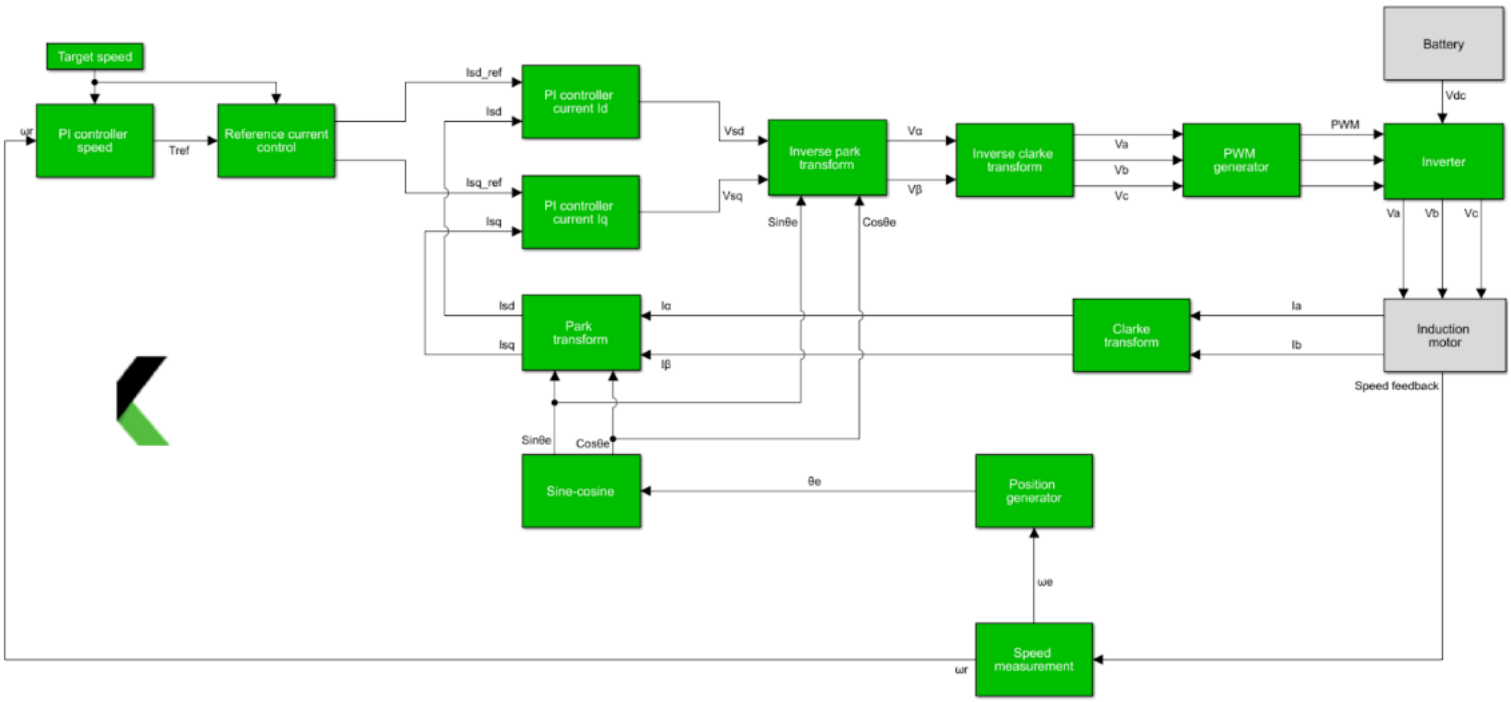


Figure 28 General overview of the system

HIL VERSION 1_1

This very simple first version was necessary to get acquainted with the dSPACE libraries. It is a first raw system that provides the engine model on the processor and the controller on FPGA.

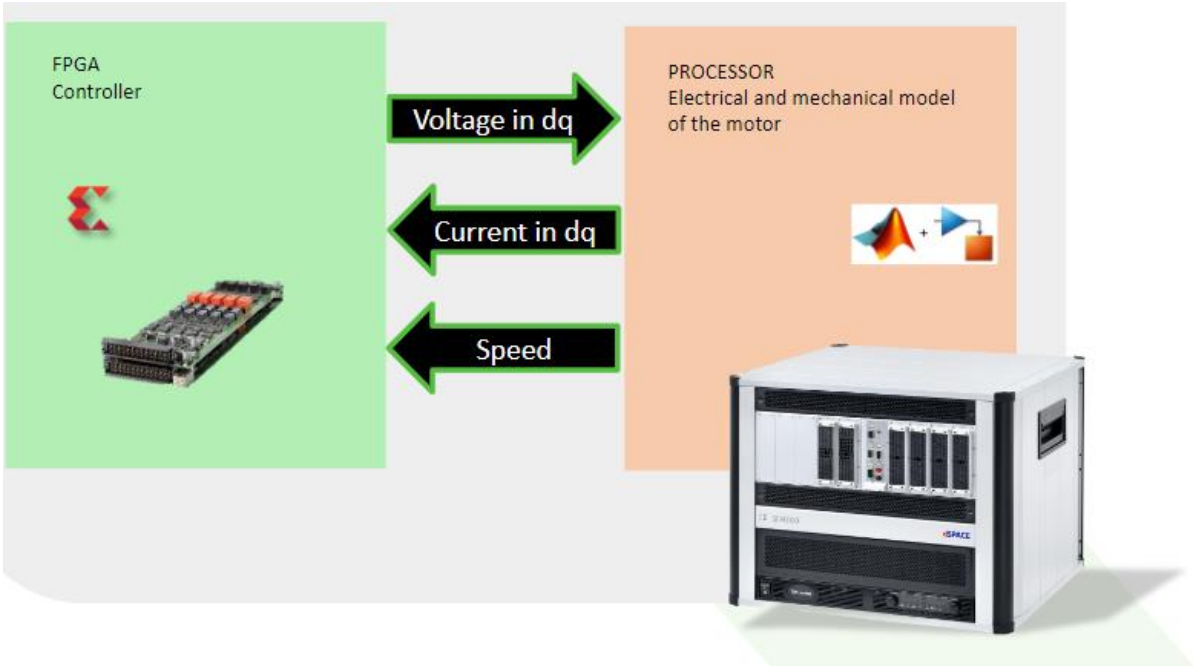


Figure 29 First version

As you can see, this model is not configured to test external components, so more than serving as HIL, this step has been necessary to test the correct functioning of the controller. It's like the project lies inside the HIL itself without being able to communicate with the outside.

The principle of operation is very simple: as you can see, both the engine and the controller work in the d-q domain. Ultimately, therefore, this model is so simple so far from the physical reality of an electric motor.

The outgoing currents from the motor are passed to the controller that compares them with a target value, performs the necessary corrections and imposes a certain voltage to the motor. The input and output of the model are well recognizable on the arrows.

HIL VERSION 1_2

The second version foresees a step forward compared to the previous one. Given that the controller in the FPGA and the engine are working correctly, the next step was to prepare the model so that it had physical outputs visible with an oscilloscope. In definitive the model does not add anything in terms of performance or accuracy of the results.

Ultimately this version provides the same precedent model, only that in the controller the currents in the domain d-q undergo the anti-transform of Clarke & Park to be seen through the AnalogOut of the FPGA. Then the oscilloscope screens will be reported with the results in which the sinusoidal currents and voltages are seen. All this is followed by the direct transforms in order to transform again the quantities in question in the d-q system and then be processed and sent to the engine.

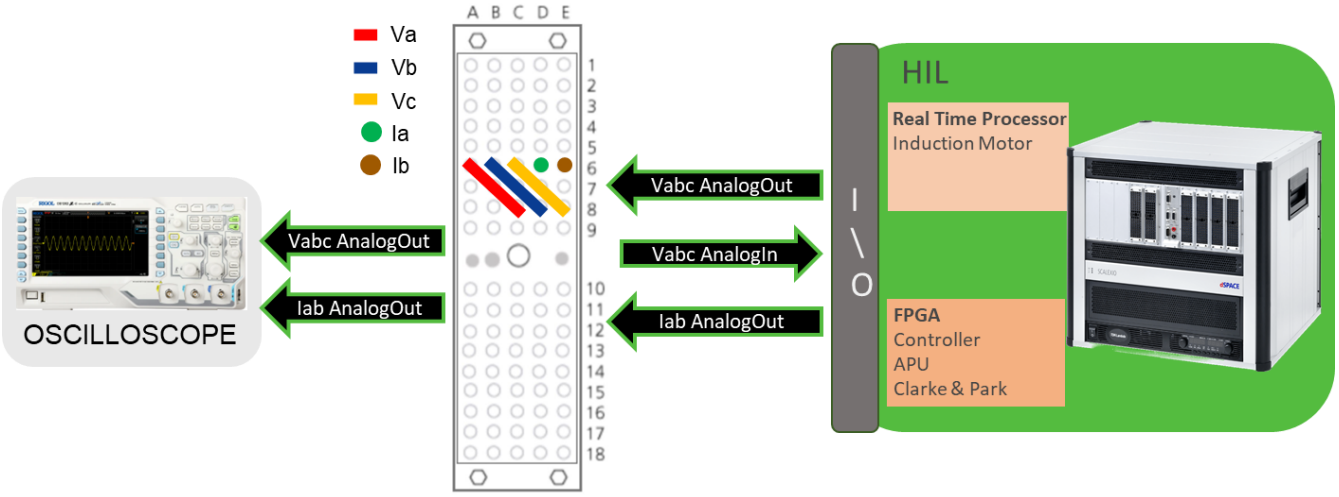


Figure 30 Second version

HIL VERSION 1_3

The third version for complexity involves the addition of another key component, the inverter. The latter is the most critical from the point of view of the management of the quantities involved due to the high frequencies. Later in the text will be explained in more detail how it was developed, but for now just know that it provides a comparison with a triangular wave with a dynamic of several orders of magnitude greater. This justifies the choice of to implement the inverter on FPGA.

The inverter works on motor control using square waves, so it was necessary to use digital channels to bring out the HIL PWM and then to bring them back inside.

The quantities handled by the model and displayed with the oscilloscope are the voltages and sinusoidal currents of the controller.

The rectangle with the colored dots between the oscilloscope and the HIL is the Hypertac port, which maps the exits from the FPGA on physical pins.

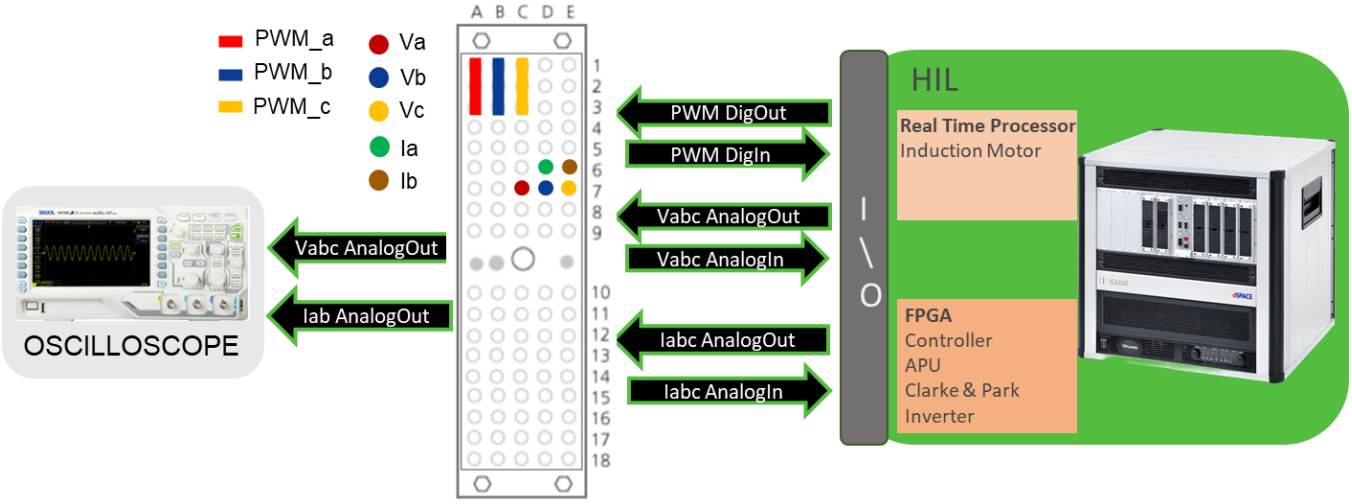


Figure 31 Third version

Before avoiding any possibility of misunderstanding it is good to make a fundamental clarification.

The final objective of the thesis is the final version, which provides for a DUT. The final model works and is correctly configured, but all the simulations, screens and considerations that will be reported refer to the HIL VERSION 1_3. From a certain point of view this need has even increased the complexity of the thesis, because, not having the controller, the latter has been modeled from scratch, thus adding degrees of uncertainty to the whole system.

HIL VERSION 1_4

This version is the definitive one and represents the most realistic case ever: the engine model is completely configured to accommodate an external control unit.
This is the real goal of the thesis and was the final step. In fact, in order to think about opening the HIL chain and putting an external controller you had to be sure that all the other components worked properly.
This final version provides the output sinusoidal currents from the motor carried outside the HIL thanks to the AnalogOut and by means of cables, which enter directly into the physical control unit. This communicates with the engine the PWM that produces thanks to digital channels.
To better understand the enormous potential of 'HIL considering them in a real business environment, we take advantage of the engines modeled to test the software "flashed" on the external control unit. To understand the correctness of the functionality of the control unit is monitored the engine and its performance. The results of these acquisitions are analyzed and, on the basis of these, it is understood if the control unit is correctly driving the engine.

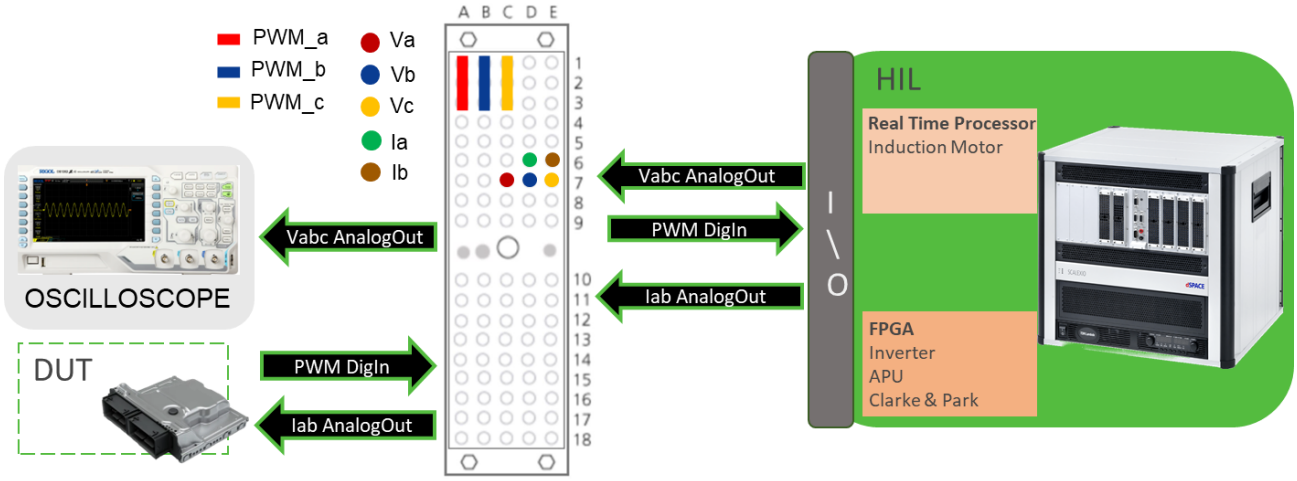


Figure 32 Final Version

Until now the project has been widely described in its entirety, considering all the components that make up the motor-inverter system. These two components were by far the most complex to study and model, while the controller and transforms required less effort.
Although the project is unique and consists of four elements that work in synergy, the thesis in question aims at the detailed description of only two of these, namely the engine and the controller. Inverter and transform will only be mentioned, in order to understand in detail the model and the principle of operation, please refer to the reading of the thesis of candidate *Luca Mutton*, who performed the complementary part of the thesis.

9.MODEL

MOTOR

Before we begin to describe the model of the engine it is necessary to contextualize it: first this has been implemented on the processor because of the dynamics of the characteristic quantities. A different speech will apply later for the controller. Proceeding in order, you will see an image of the main screen of the project seen in Simulink:

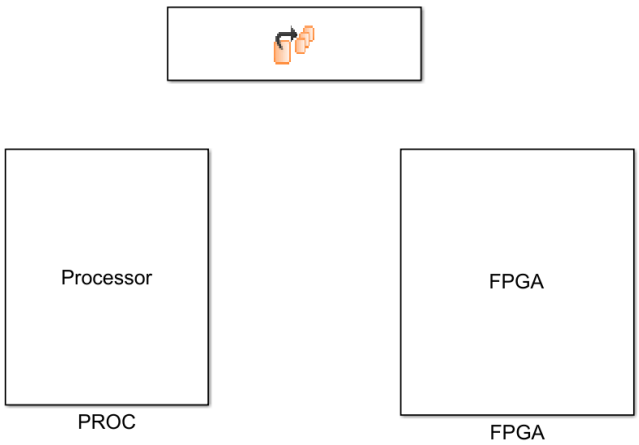


Figure 33 MATLAB overview (screen from Simulink)

In general, all the models will be presented starting from the highest level up to the detail of the elementary blocks so as to create a kind of path to read the model. This premise aims to present the facts in the clearest possible way, but logically in the modelling phase we proceeded exactly the opposite, then modeling first the most elementary blocks and then climbing complexity integrating the various parts.
Going into detail, this first screen presents the first big distinction, that is the components simulated on the processor and those implemented on the FPGA to be simulated in real time. As said, the engine has been modeled entirely on a processor, so to begin to explain the model you have to enter the left block.
Entering the left block, the model looks like this

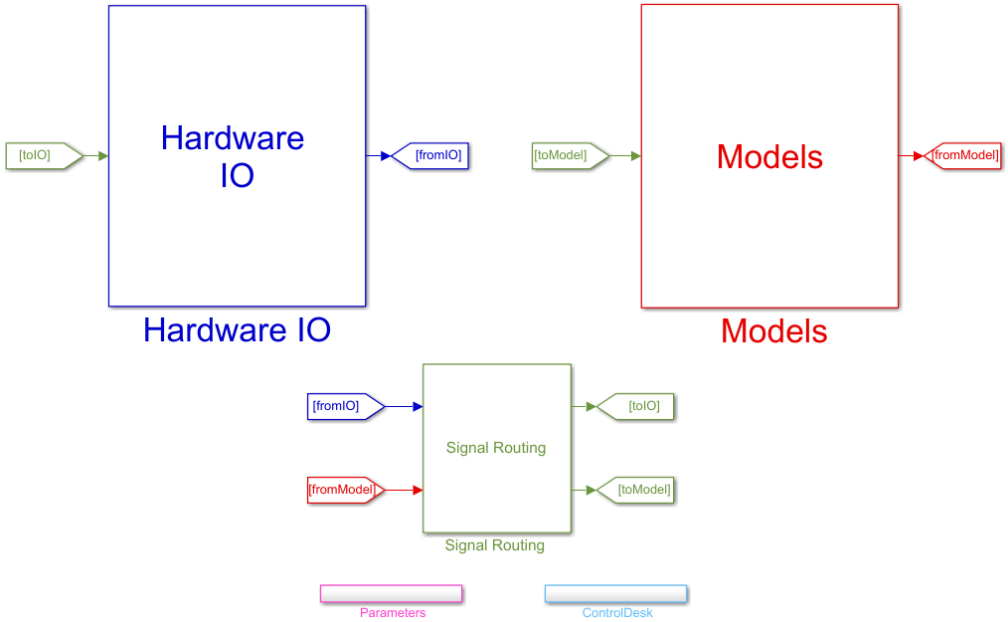


Figure 34 Processor general overview (screen from Simulink)

For the 4 versions it was decided to implement a standard always the same so as to make all the models clear and readable. In detail the blocks have a precise function.

- IO hardware: this is the block that contains all the physical quantities that are "passed" between the processor and the FPGA. Inside the block is further divided between what is input of the FPGA (and that then exits from the processor) and what instead exits from the FPGA (which then falls into the processor)

Figure 35 FPGA IO (screen from Simulink)

- Signal routing: this block contains all the variables present inside the model, both those exchanged between processor and FPGA, and those inserted directly as calibrations by means of parametric quantities *(FromUserInput)*

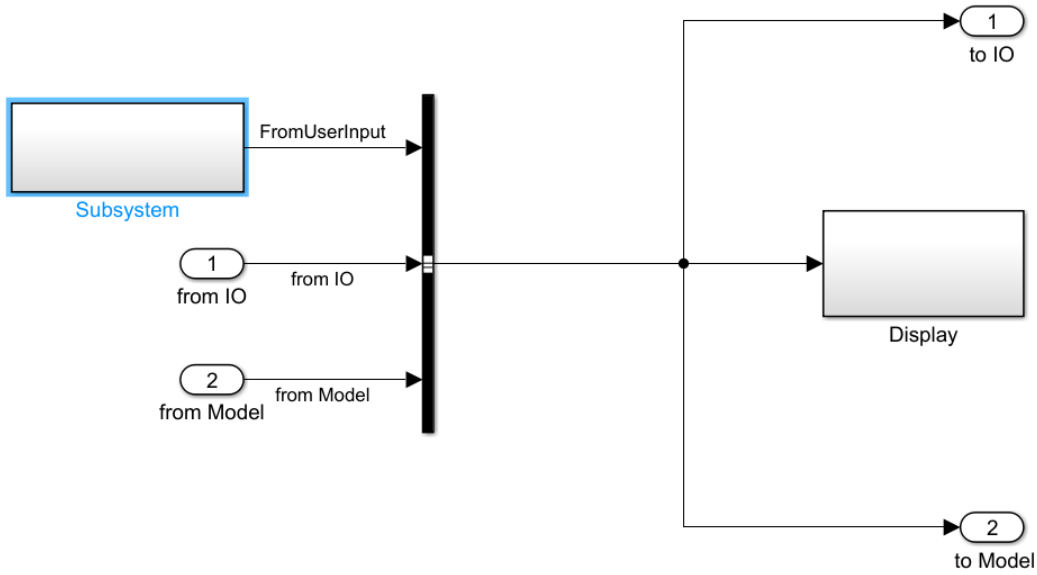


Figure 36 Signal routing (screen from Simulink)

- Parameters and Control Desk: The blocks contain all the parameters of the model, distinguishing in particular the parameters of the motor (resistors, inductances, etc.), those of the controller (all the typical gain of a PI) and those that are passed to Control Desk, which will be discussed in detail later.

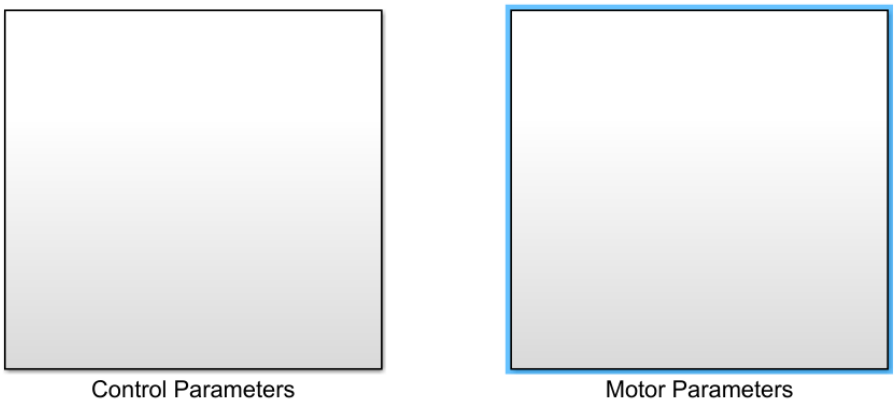


Figure 37 subsystems with the parameters (screen from Simulink)

- Models: is undoubtedly the most substantial part of the script. It includes all the part of the model that describes the components that are implemented on the processor, then the motor and the blocks responsible for calculating the electrical and mechanical quantities.

Now let’s analize how we modeled and induction motor on simulink first from an higher point of view anf than going always more in depth:

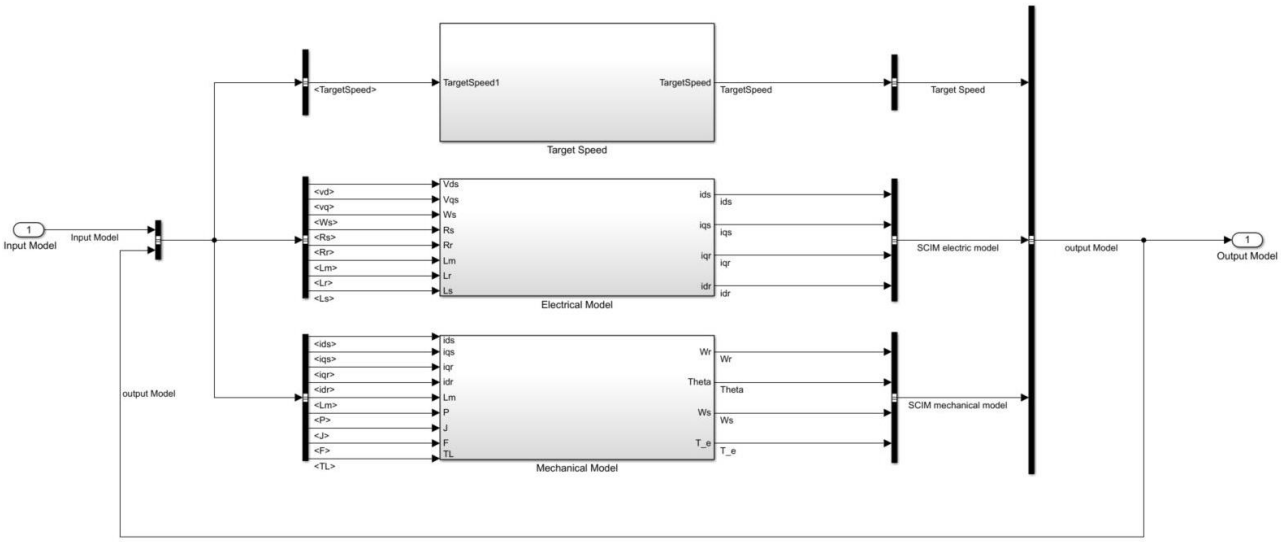


Figure 38 Motor general overview (screen from Simulink)

Before starting to report and explain all the block we want to fix that all the computations are referrred to d-q reference frame.
This is how we decided to divide the whole “motor” into three main blocks:

Target Speed:

This block serves simply to bring back the target speed inside the engine model.

Electrical model:

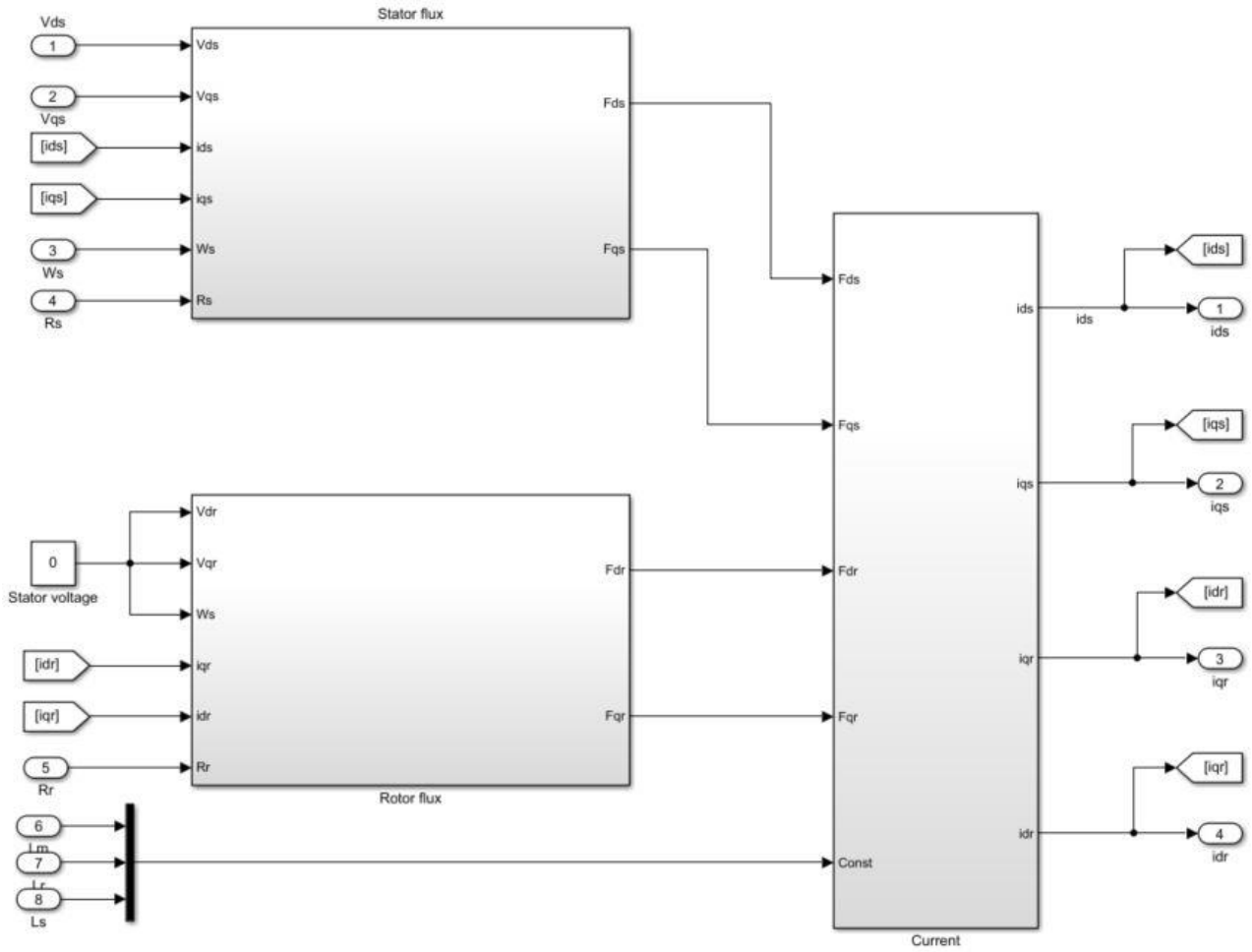


Figure 39 Electrical model of the motor (screen from Simulink)

This big subsystem comprends all the calculations needed to get the current representing the actual speed. This currents will be useful later in the next block.
For now let's focusing on the first thing that can be noticed:

$$V_{dr} = 0$$
$$V_{ds} = 0$$

This is due to the motor istelf: we have said that the rotor bar are short circuited, and so we imposed a null voltage.

Stator Flux

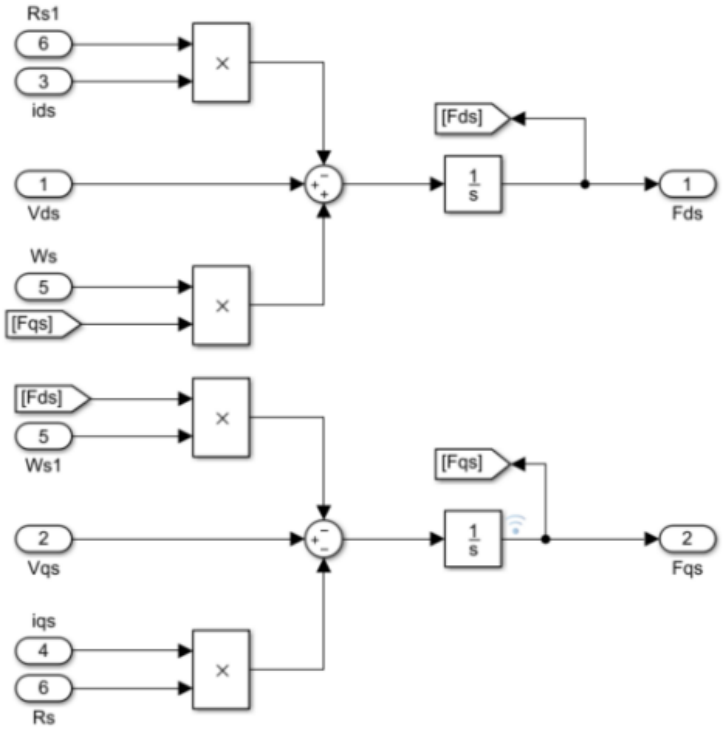


Figure 40 Stator flux calculations (screen from Simulink)

This script implements the following equations:

$$\frac{d}{dt}\begin{bmatrix}\lambda_{ds}\\ \lambda_{qs}\end{bmatrix} = \begin{bmatrix}V_{ds}\\ V_{qs}\end{bmatrix} - R_s\begin{bmatrix}i_{ds}\\ i_{qs}\end{bmatrix} + \omega_s\begin{bmatrix}0 & -1\\ 1 & 0\end{bmatrix}\begin{bmatrix}\lambda_{ds}\\ \lambda_{qs}\end{bmatrix}$$

Where:

- λ_{ds} = stator linked fluxes d-ref
- λ_{qs} = stator linked fluxes q-ref
- R_s = stator resistance [Ω]

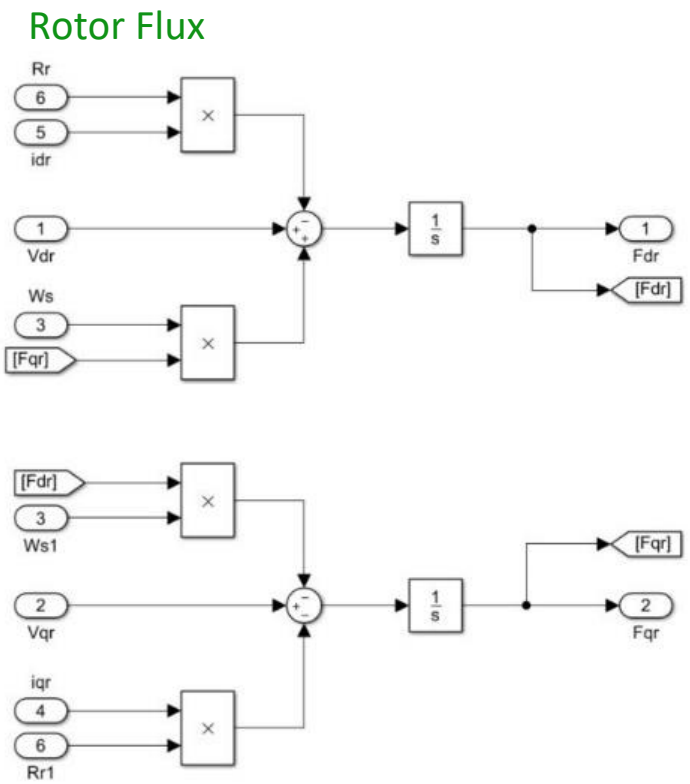


Figure 41 Rotor flux calculations (screen from Simulink)

This script implements the following equations:

$$\frac{d}{dt}\begin{bmatrix}\lambda_{dr}\\ \lambda_{qr}\end{bmatrix} = \begin{bmatrix}V_{dr}\\ V_{qr}\end{bmatrix} - R_r\begin{bmatrix}i_{dr}\\ i_{qr}\end{bmatrix} + \omega_s\begin{bmatrix}0 & -1\\ 1 & 0\end{bmatrix}\begin{bmatrix}\lambda_{dr}\\ \lambda_{qr}\end{bmatrix}$$

Where:

- λ_{dr} = rotor linked fluxes d-ref
- λ_{qr} = rotor linked fluxes q-ref
- R_r = rotor resistance [Ω]

Just to be clear, in the script it has been using F_{ds} with the same meaning of λ_{ds}. The choice of using F instead of λ is just to lighten the notation.

Currents

The next script implements the following equations:

$$\begin{bmatrix} i_{ds} \\ i_{qs} \\ i_{dr} \\ i_{qr} \end{bmatrix} = \left(\frac{1}{L_m^2 - L_r L_s}\right) \begin{bmatrix} -L_r & 0 & L_m & 0 \\ 0 & -L_r & 0 & L_m \\ L_m & 0 & -L_s & 0 \\ 0 & L_m & 0 & -L_s \end{bmatrix} \begin{bmatrix} \lambda_{ds} \\ \lambda_{qs} \\ \lambda_{dr} \\ \lambda_{qr} \end{bmatrix}$$

Where:

- L_s = Stator Inductance [H]
- L_r = Rotor Inductance [H]
- L_m = Magnetizing Inductance [H]

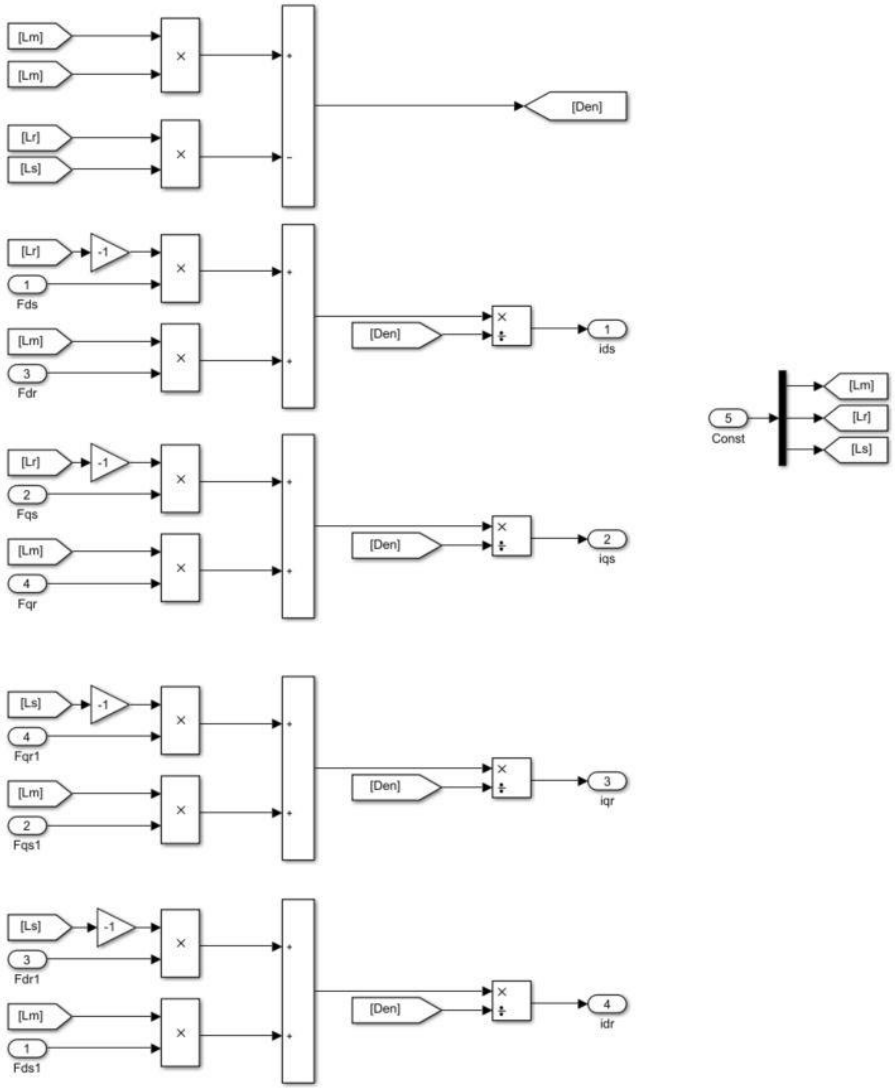


Figure 42 Currents calculations (screen from Simulink)

This last script ends the Electrical subsystem of the motor. Starting from the physical parameters, the voltage, and the speed of the magnetic rotating field we've computed the current in d-q frame needed to control the motor.

Mechanical model:

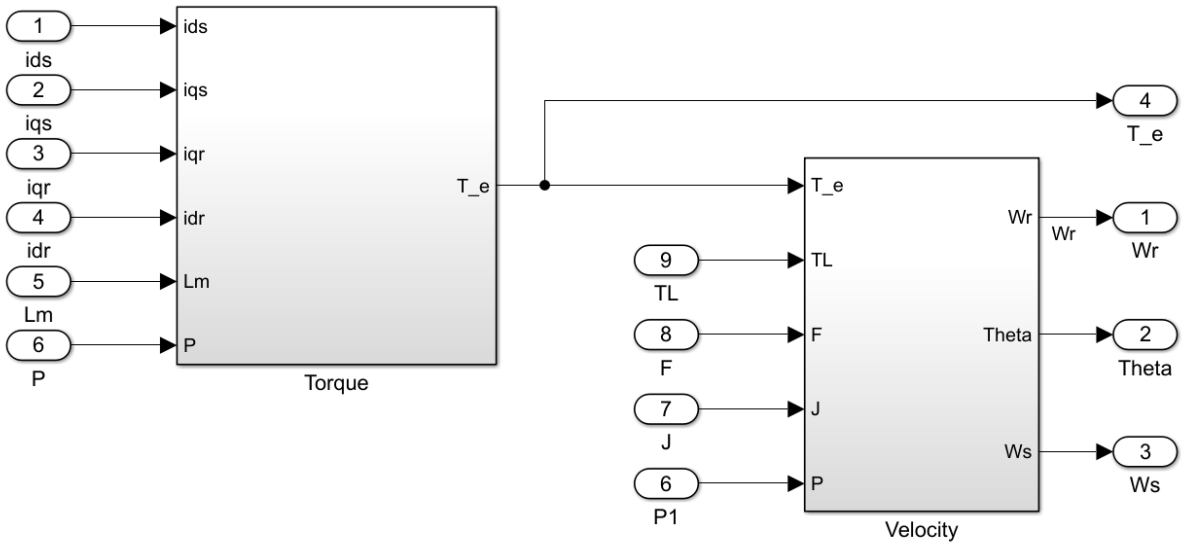


Figure 43 Mechanical model of the motor (screen from Simulink)

This is a general overview of the working principle of the motor controlled in current: starting from the supplying current we've been able to compute the mechanical parameters typical of a motor.

Torque

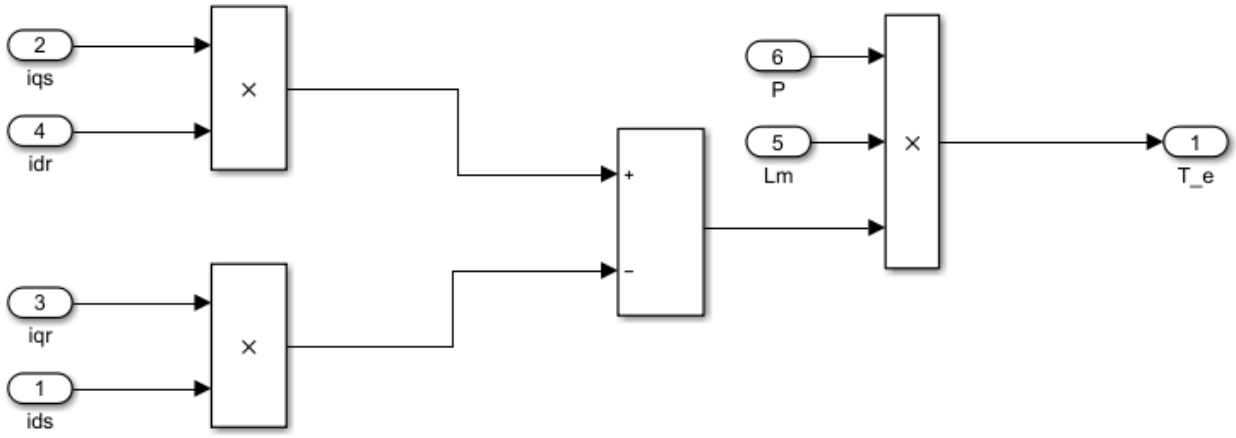


Figure 44 Torque calculations (screen from Simulink)

This Simulink's script implements the following equation:

$$T_e = P * L_m(i_{qs} * i_{dr} - i_{qr} * i_{ds})$$

Where:

- T_e = Electromagnetic Torque [Nm]
- P = Pole Pairs
- L_m = Magnetizing Inductance [H]
- i_{qs} = stator current q-ref [A]
- i_{ds} = stator current d-ref [A]
- i_{qr} = rotor current q-ref [A]
- i_{dr} = rotor current d-ref [A]

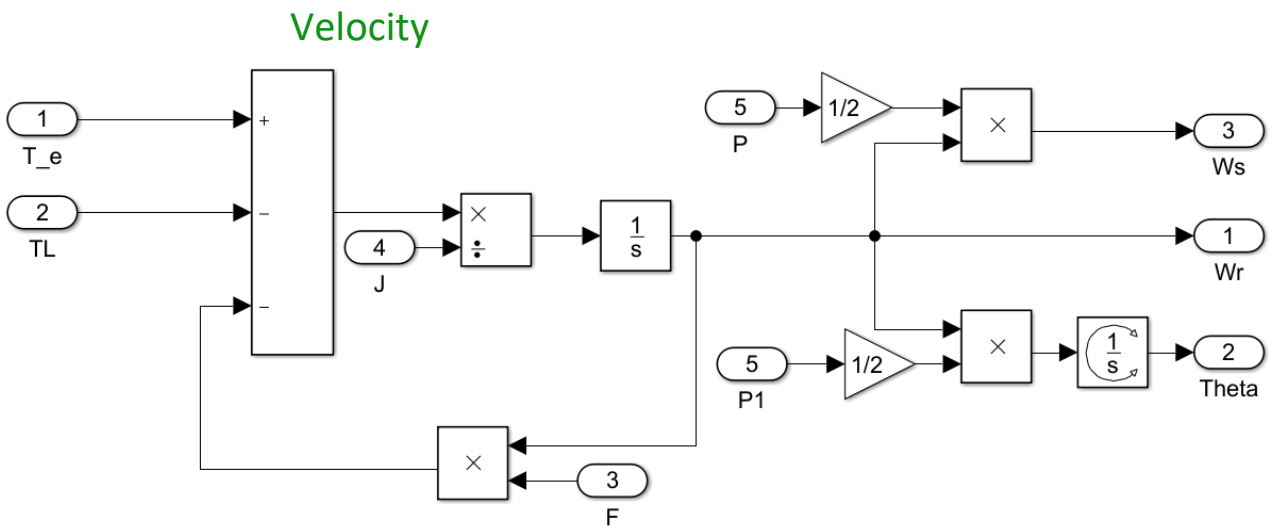


Figure 45 Velocity calculations (screen from Simulink)

This block implements the following equations

$$\begin{cases} \frac{d}{dt}\omega_m = \frac{1}{J}(T_e - T_l - F\omega_m - T_m) \\ \frac{d\vartheta}{dt} = \omega_m \end{cases}$$

Where:

- T_e = Electromagnetic Torque [Nm]
- T_l = Load Torque [Nm]
- T_m = Rotor shaft Torque [Nm]
- J = Moment of inertia [Kg*m^2]
- F = Viscous Friction [Nm*s/rad]
- ϑ = rotor angular position [deg]
- ω_m = Mechanical speed of the rotor [rpm]

CONTROLLER

Having removed the first argument of the thesis, we continue the explanation of the project following the flow described in the figure of the previous paragraph.

The engine has now been described. This receives three-phase voltages from the inverter, which for now is only mentioned, and generates two types of output: a mechanical, then the mechanical speed of the rotor and the torque delivered, and one of an electrical nature, or currents. The two outputs together are sent to the controller, who performs its corrections.

Looking in more detail at the overview you can see how before arriving in the controller the three-phase currents are transformed into the d-q references. This choice was made in order to make the control more precise and simple to implement. However, although the transformers are between the motor and the controller, these will be explained only later, as they are not central to understanding the operation of the controller.

Downstream of this brief introduction it is however necessary to remember that the choice to implement a simulated controller was forced by the fact that during the thesis path it was not possible to access a real external control unit. The goal was to configure an engine that could host a real DUT, but because of the cost and the need to close the loop it was decided to simulate one in FPGA. Nevertheless the project was executed according to its original guidelines, in fact the engine was implemented in order to have the IO channels configured to really host an external physical ECU.

Ultimately the controller has added an element of difficulty to the project, since there is not already ready and validated was created from scratch. Below is all the controller modeling.

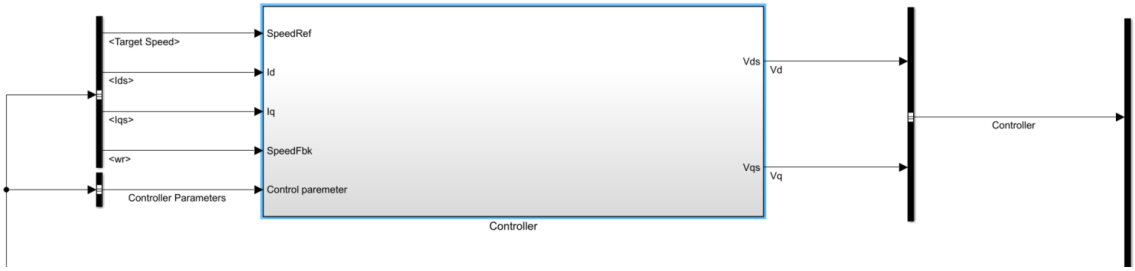


Figure 46 Controller general overview (screen from Simulink)

Later will be describe how it works in detail, but for now can noticed that the inputs of the controller are a series of physical parameters, some constant typical of the controller and most of all, the actual working conditions that will be compared with the desired values.

The output indeed is the voltage (always in d-q reference frame) that shall be applied to guarantee the desired performances. This voltage is sent to the inverter which will generate the tree-phase voltages.

Let's see what's inside the outer subsystem:

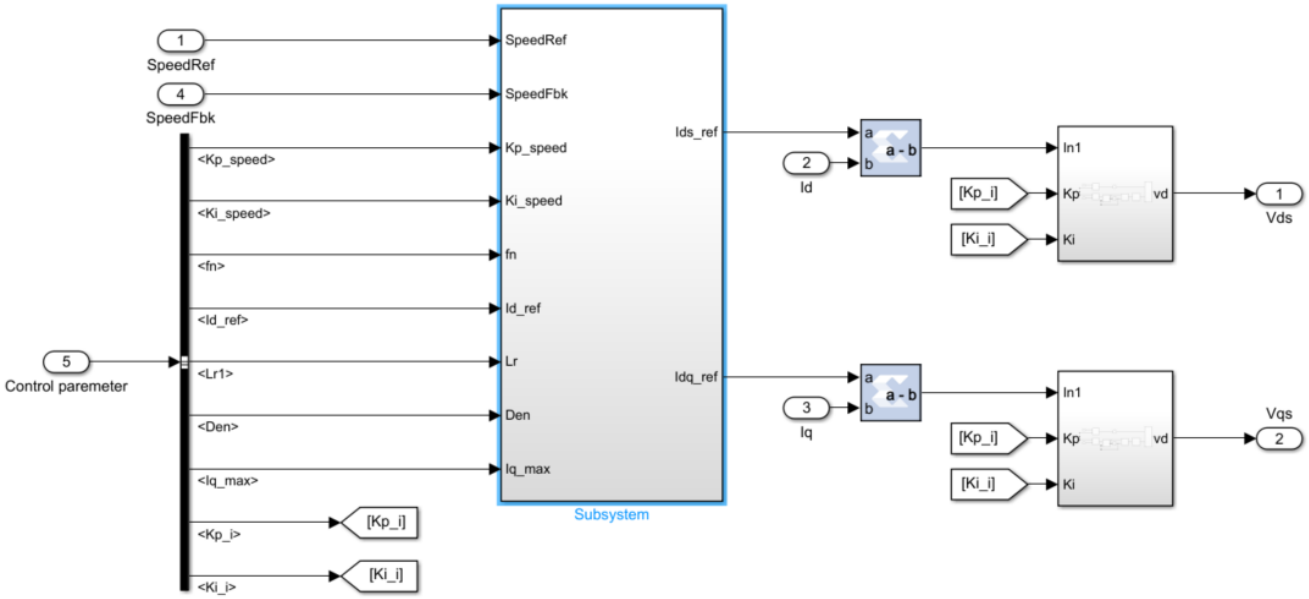


Figure 47 Controller (screen from Simulink)

Going more in depth, the first block computes the currents starting from the speeds.

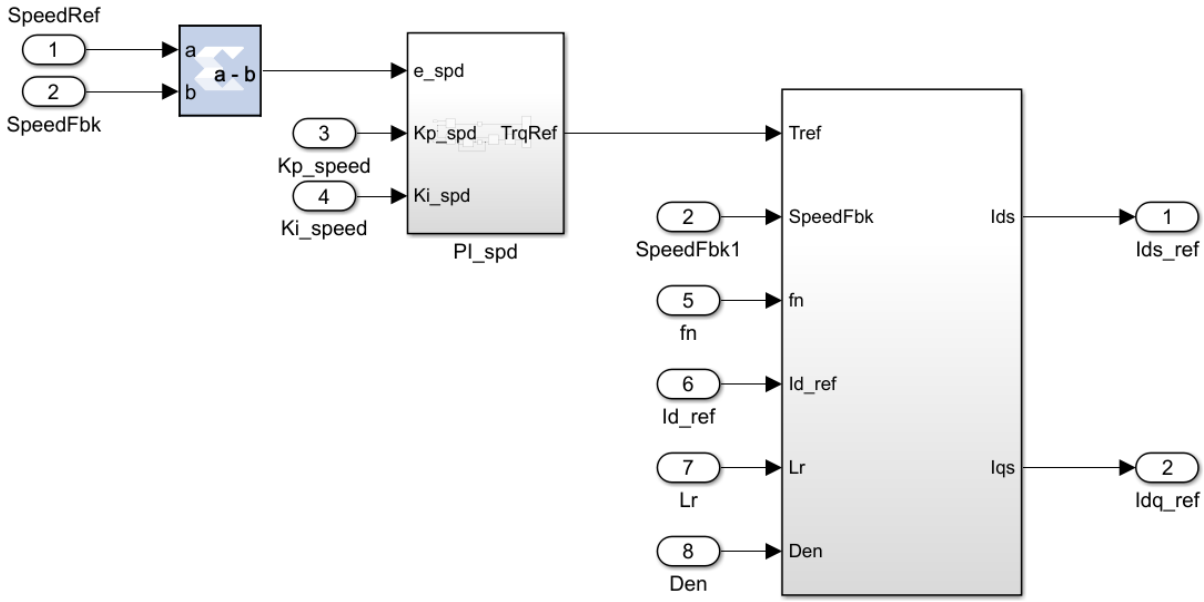


Figure 48 Currents calculations (screen from Simulink)

The first thing to do is to compute the difference between the actual speed and the desired one. This is important because it expresses how invasive shall be the action of the controller.

$$e_{speed} = \omega_{ref} - \omega_{fdbk}$$

Where:

- e_{speed} = Difference between the actual speed and the desired one
- ω_{ref} = Desired speed [rpm]
- ω_{fdbk} = Actual speed [rmp]

This error becomes smaller while the simulation goes ahead. Theoretically it should become equal to zero at the steady state, when the commanded speed should be equal to the desired one.

The result of this computation goes in the PI controller:

- The term P has to do with the difference between the actual difference between the target speed and the feedback value coming from the plant. Depending on the amount of the difference the gain factor K_p is tuned: if the error is large, then the gain will be large
- I term indeed keeps in mind the past values of the error ($\omega_{ref} - \omega_{fdbk}$) and integrates them over the time. Practically speaking, if a residual error remains, it is compensated by the controller adding a term related to the past values. When the error is erased, the integral action stops increasing and goes to zero. K_i is the typical gain constant of this term.

To get the correct control action become important the correct tuning of the constant. They are strictly related to the application because of their influence on the dynamic response of the system. They influence the stability of the system and the response to external stimuli. This technic is called “loop tuning” and consists on adjusting the control action by working on the value of the PI controller’s parameter. In general, is not easy to tune these values since the final application may requires a short transient but also to be very smooth and with a small overshoot.

Coming back to the PI controller, the overall control function has the following equation:

$$u(t) = k_p e(t) + k_i \int_0^t e(t)\, dt$$

Where:

- $u(t)$ = output of the controller
- $e(t)$ = error between the set point and the actual measure
- k_p = proportional gain constant
- k_i = integral gain constant

Below will be reported how has been implemented this equation in the model.

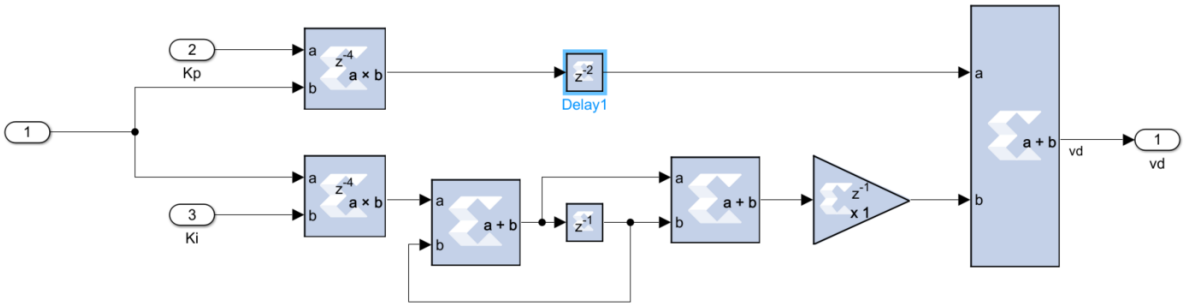


Figure 49 PI equation (screen from Simulink)

The controller has been implemented in FPGA, and so all the computations have been done with discrete signals. The integral is not defined with non-continuous functions, and so it was done in a different way: sum with a delay has been implemented. This is more or less what an integral does also with coninuous signals. Following this idea, it has been performed a sum by adding to the actual value the previous one. This can be done thanks to the delay's block.

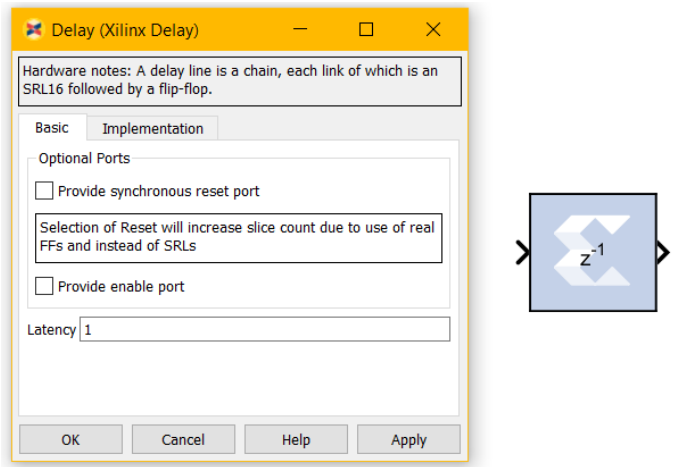


Figure 50 Delay block (screen from Xilinx library)

The block receives an input signal, keeps it for a time interval equal to latency and returns it as output.

In the FPGA, every number is represented as a binary with n-bits, and all the operations are performed sequentially. One of the biggest difficulties that has been to faced off is to perform the operations between the correct numbers: some operations, such as multiplications, require a longer processing time than a simple sum, so the output of a block of a product will take longer to be processed.

This explains why a delay block appears in the upper branch of the diagram: the final sum of the PI must be made between two numbers that correspond to the same physical instant of time. The delay does not add anything from the controller modeling point of view, it simply serves to make the two addends coming from the two parallel branches arrive at the same time at the final sum, which otherwise would arrive with different latencies since the time difference required to process the two incoming signals.

In the specific case described above, it is clear that the two branches of k_p and k_i require different processing times: the first has only a multiplication, while the second is characterized by a product and subsequently an integral.

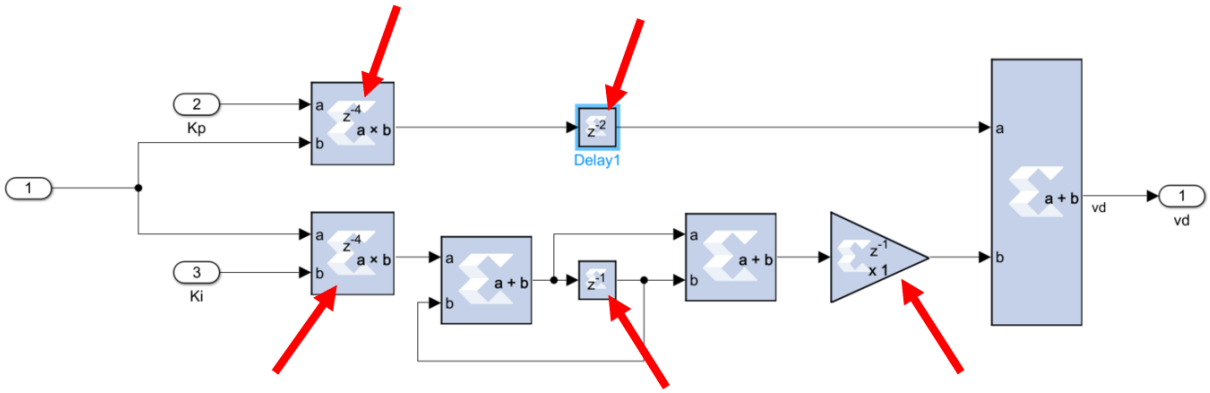


Figure 51 Delay considerations

The sum of a and b must be between elements belonging to the same physical time. It was necessary to add the highlighted delay block. Then the concept of timing analysis will be explained better, for now just note that:

- the upper branch (k_p) has only one multiplication which is carried out within a latency of 4 ticks
- the lower branch (k_i), on the other hand, has a latency of 4 on the first multiplication, 1 on the integral and then one on the last gain.

In the end, the operation of $k_p$requires 4 ticks, that of $k_i$requires 6. This would lead to a consequence: at the 8 tick of the processor the sum would have already processed twice $k_p$and only one k_i. This would lead to a Vd (in this case) calculated with parameters of the PI

controller belonging to different time instants. It's as if the $k_i$is still the instant t0, whereas $k_p$belongs to (t0+Δt). To correct this problem that would lead to serious errors in the results was added the delay of 2 tick so as to arrive at the same time the two addends of the sum.

Going back to talking specifically about the model, starting from the difference between the feedback speed and the target speed, it was possible to calculate the torque to be imposed to reach the desired speed.

Below is the last subsystem of the controller, the one in charge of calculating the current required by the motor to reach the target speed starting from the Torque. The following equations have been implemented:

$$i_{qs} = \frac{T_{ref} \cdot L_r}{Den}$$

$$Den = i_{ds\_ref} \cdot P \cdot L_m^2$$

Where:

- i_{qs} = stator current q-ref [A]
- i_{ds_ref}= stator reference current d-ref [A]
- T_{ref} = Reference torque [Nm]
- L_r = Rotor Inductance [H]
- L_m= Magnetizing Inductance [H]
- P = Pole pairs

For i_{ds} the situation was slightly more complicated as the system implements two possible ways based on the feedback speed. The idea is that:

- If $|\omega_{fdbk}| \leq \omega_{rated}$
  if the actual speed is lower than the nominal one, the delivered current is made to saturate to a calibratable value to increase the rotor rpm and reach the target speed

$$i_{d_ref} = i_{ds}$$

- Else

  in the opposite case, when the current delivered is too high, then this is divided by a value proportional to the current speed

$$i_{ds} = \frac{i_{ds\_ref} \cdot \omega_{reted}}{|\omega_{fdbk}|}$$

$$\omega_{reted} = 2\pi f_n$$

Where:

- i_{ds} = stator current d-ref [A]
- i_{ds_ref} = stator reference current d-ref [A]
- ω_{fdbk} = Mechanical speed of the rotor [rpm]
- ω_{reted} = Rated speed of the motor [rpm]
- f_n = nominal frequency [Hz]

The trapezoid block represents a switch implemented in FPGA. The first line expresses the condition that allows one of the two cases to occur. Based on the result of the first channel, the second or third is activated. In the specific case, when the Boolean condition of the inequality gives 0 as a result, therefore the nominal speed is greater than the current one, the d0 channel is implemented which saturates the current to a calibratable value. If instead the result is 1 it means that the feedback speed is greater than the nominal one, therefore the current supplied by the controller is scaled

The block that implements the following equation in the Xilinx library represents the exponential.

$$a\Lambda^b$$

Since in this library the division does not exist, a different solution had to be found. The inputs of the block are the number (a) and the exponent (b). In the case in question, therefore, the feedback speed was taken and raised to -1. By doing so, the problem of division was solved.

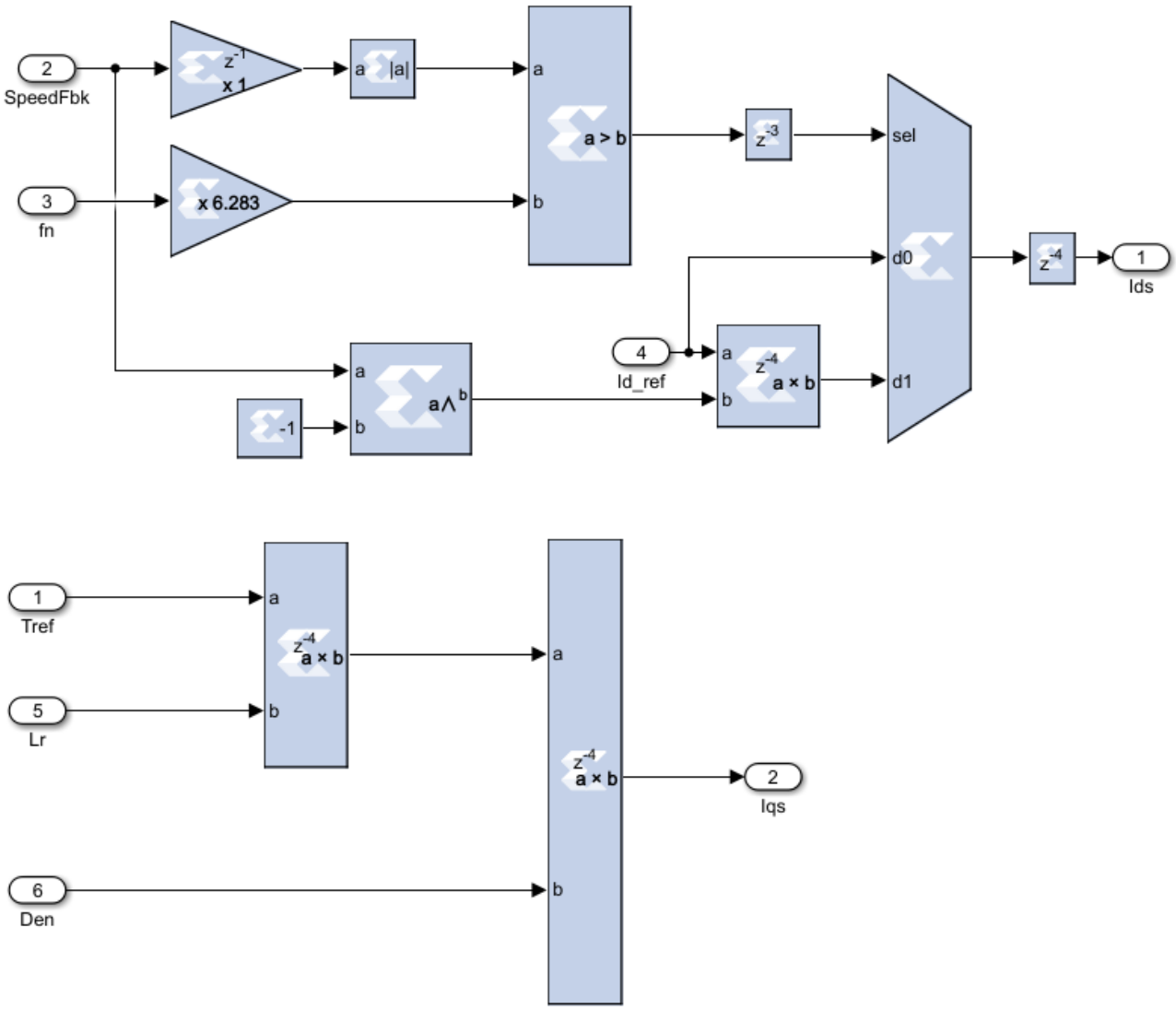


Figure 52 Current calculations (screen from Simulink)

The two currents just calculated in this way are then compared with the actual ones. The physical idea is that the reference currents are those necessary to reach the target speed. By putting them as input of a PI controller it was then possible to calculate the final speeds commanded by the controller just comparing the actual currents and the reference ones. To better understand what you are referring to we report the same image presented above highlighting the blocks involved in the calculations:

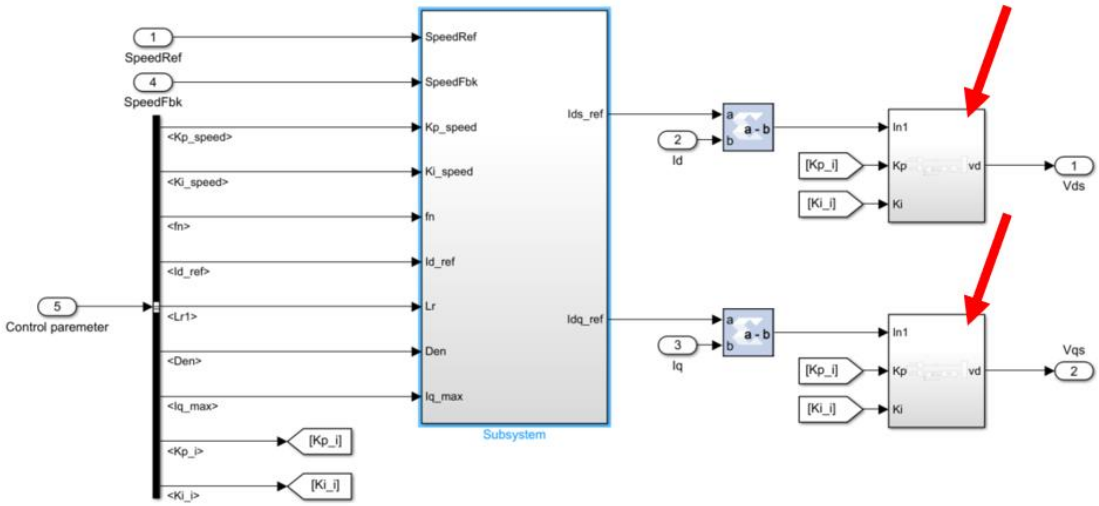


Figure 53 Recap overview

Vd and Vq output from the model are the outputs imposed by the controller such that the current speed converges to the target speed.

APU

The APU block was added only in the advanced stages of the project. The need to insert it arose when we noticed the difficulty of the model to follow the target speed when it had changed rapidly. Until the block was added, the electric angle was calculated in the processor and then passed to the FPGA to calculate the inverse transforms of Clarke & Park. The problem was the interface between the processor and the FPGA.
Making a quick calculation It is immediately noticeable that the accuracy of the angle passed to the inverses of Clarke and Park is not sufficient in case the speed changed too quickly. Sampling at the ms could lose intermediate values between one tick and another.
To better understand what you are talking about, just think of a numerical example imagining you have a rotor that rotates at 10000RPM. By converting the number to RPS, turn to the second, and then turn to the millisecond, you immediately notice that in one millisecond the rotor completes about 1/6 turn. In other words, it takes 6ms so that the rotor completes a rotation, which results in an angle of 2π divided into only 6 steps. This is a very low precision because you risk losing acquisitions related to intermediate angular positions at the 6 steps recognized.
For this reason, the calculation was later inserted into the FPGA. It is an integral for the calculation of the electric angle starting from the speed of the rotor, but being carried out directly in FPGA allows a more precise calculation of the angle and therefore a better operation of the motor in response to speed changes.
Below are reported the screen of the model, but, as for the next components, for a more detailed description please refer to the reading of the thesis of *Luca Mutton*

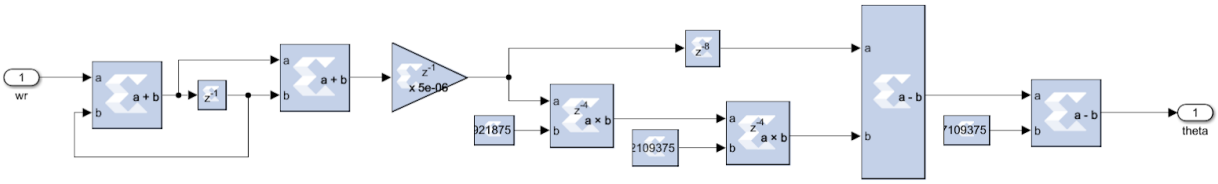


Figure 54 APU calculations (screen from Simulink)

CLARKE & PARK TRANSFORM

In order not to lose sight of the overall vision of the project, a block diagram of the whole model is reported, so as to always have in mind the point you have reached with the discussion and where you must arrive.

Treated the engine model and the controller now you need to delve into the remaining sub-systems. Clarke & Parke transform blocks and anti-transforms are currently being examined.

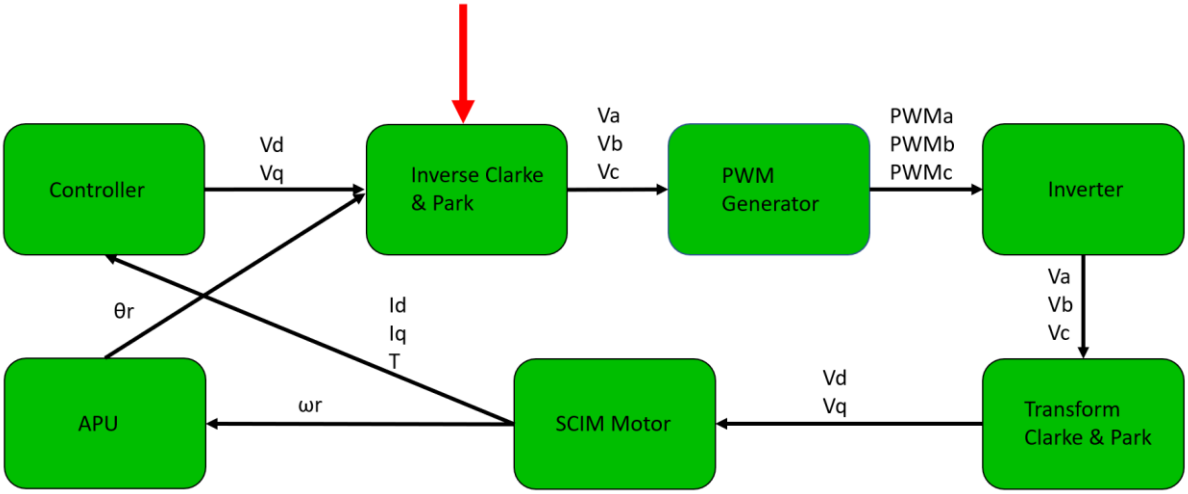


Figure 55 Global system overview

It is however careful to remember that in this writing they will only be mentioned. For a detailed and exhaustive discussion of the modelling, please refer to the thesis of the candidate *Luca Mutton*, graduated in the December session at the Polytechnic in Turin with the complementary part of this thesis. Here is a brief summary of what emerges in its report in order to allow the complete modeling of the motor + inverter to then be able to explain the final results.

The following image is a snapshot of the block as it is implemented inside the model. The input and the output are clearly visible.

Figure 56 Inverse Clarke & Park general overview (screen from Simulink)

Inside the block there are two sub-systems in charge of computing the transformation. Here it is depicted the inverse, but the direct transform works in the very same way.

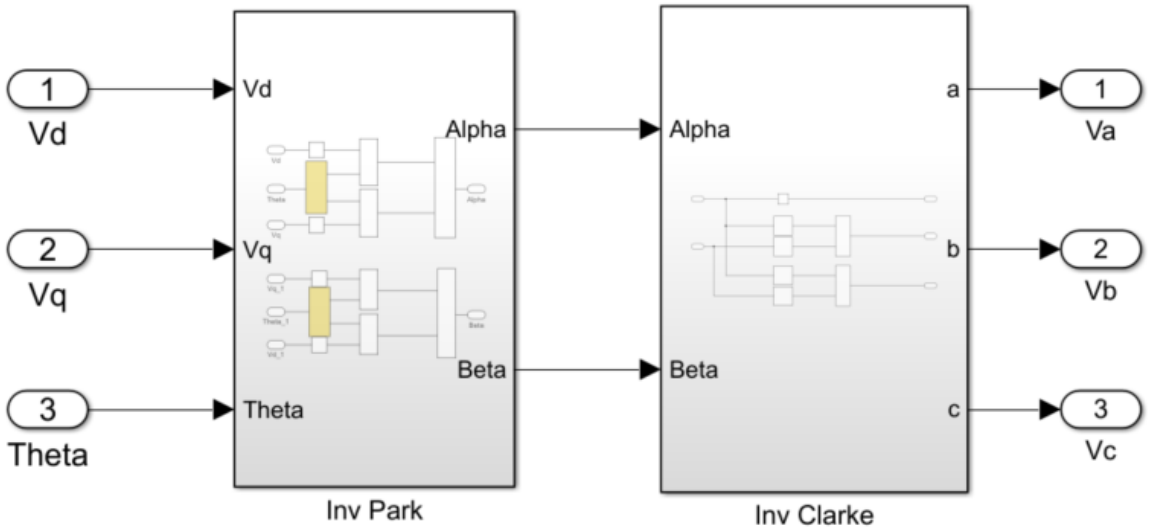


Figure 57 I/O of Clarke & Park inverse transform (screen from Simulink)

The inverse of Park transforms the voltage in the d-q reference frame coming from the controller into a variable expressed in the rotating frame alpha-Beta. The theta angle is the angle that expresses the rotational speed of the rotor. It is calculated from a small block in FPGA that will be presented later. This is how the equations have been implemented:

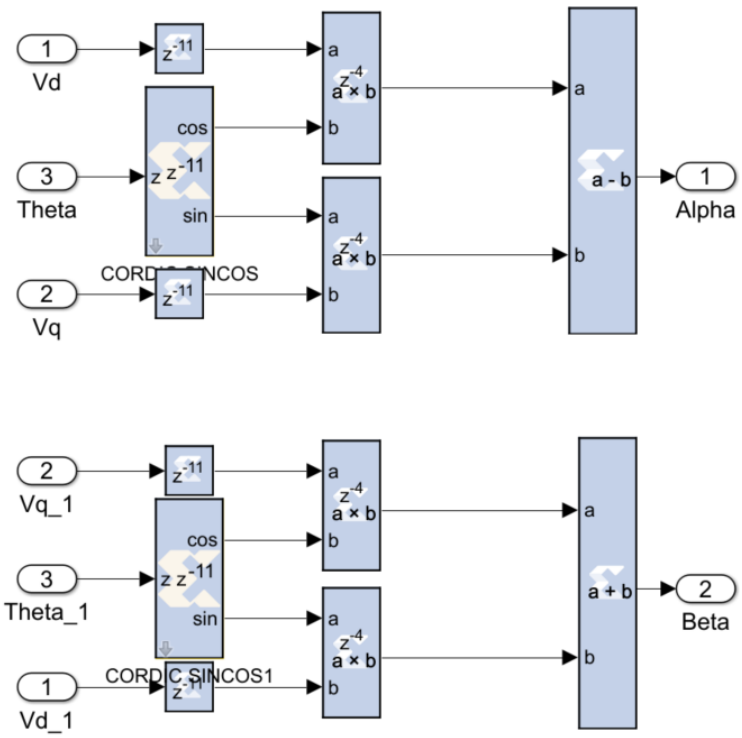


Figure 58 Inverse ok Park (screen from Simulink)

From a computational point of view, you can immediately see how these are the most complicated blocks to implement. This can be seen from the large latencies that have been put in individual operators.
In conclusion, the inverse of Clarke converts the obtained result in the three-phase system.

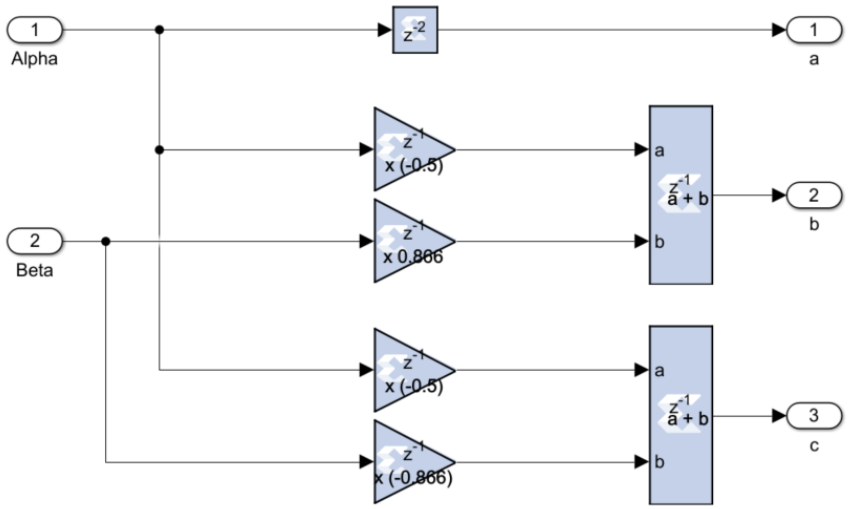


Figure 59 Inverse of Clarke (screen from Simulink)

The output of the two anti-transformed do not add anything from the point of view of the performance of the model. These are simply the voltages that the controller has calculated to impose on the motor in order to reach the target speed, expressed in a sinusoidal system to be exploited by the inverter rather then in d-q.

PWM GENERATOR

Now, let's see why the V_{abc} have been calculated.
First, a brief reminder of the status of the described works and those missing.

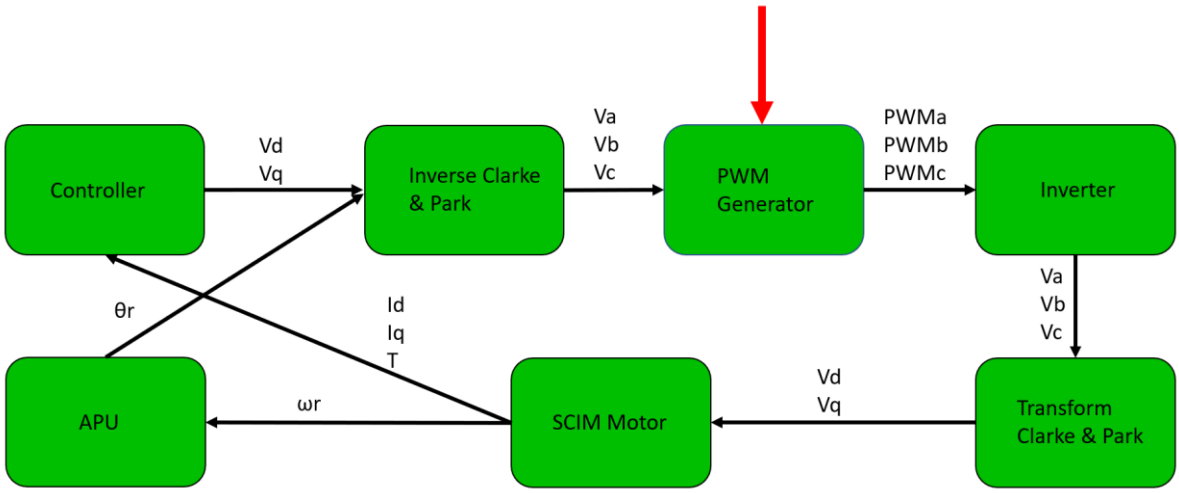


Figure 60 Global system overview

In many applications, the possibility to control a variable that affects the system is required. The argument analyzed is not an exception: the control of the motor speed and its torque is carried out by the current that is delivered by the controller. For this reason, it is necessary to change the amplitude and frequency of this variable. This is possible thanks to the use of PWM (Pulse Width Modulation).

Below is a diagram of how square wave creation was implemented in the model. Briefly, these are the result of a logical front line between an input signal and a triangular wave. The higher the frequency of the triangular wave, the more detailed will be the output of the system. Just to give you an idea, the order of magnitude of the triangular waves is an order of magnitude larger than the input signal. Here the reference signal has been modeled trough a counter which rises until a precise value and then it is immediately resets to 0.

V_{abc} input is the three-phase voltage supplied by the controller in order to obtain the desired speed. This is scaled by a factor and compared to the triangular wave. Whenever the voltage is greater than the value of the triangular wave, the output is equal to 1. It is evident that the alternation of 1 and 0 generates a square wave. This is done for all three three-phase voltages.

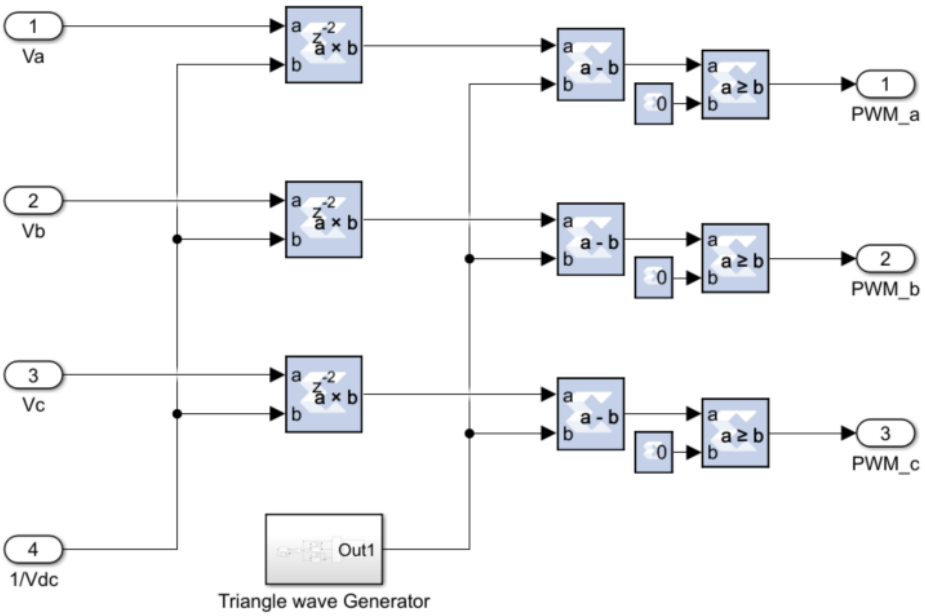


Figure 61 PWM generation (screen from Simulink)

INVERTER

Generated PWMs are used in the inverter. This component has been extensively modeled in the thesis of *Luca Mutton*, therefore, in this case, for any further clarification please refer to the reading of his work.
In a very short way here we report a series of images depicting the model. This first script summarizes how the input square wave is transformed into a very noisy sine wave:

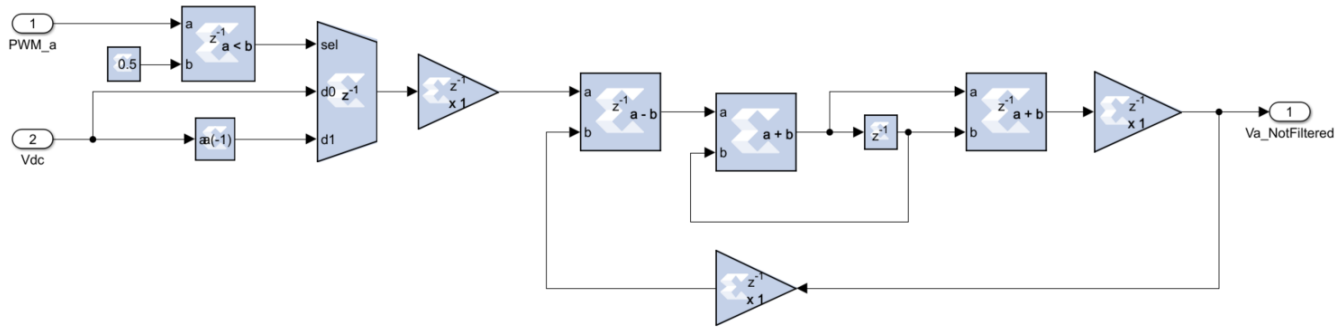


Figure 62 Sine from PWM calculations (screen from Simulink)

After the computation, a filter has been implemented to clean up the signal. It is an RC filter which computes the mean value of the incoming signal.

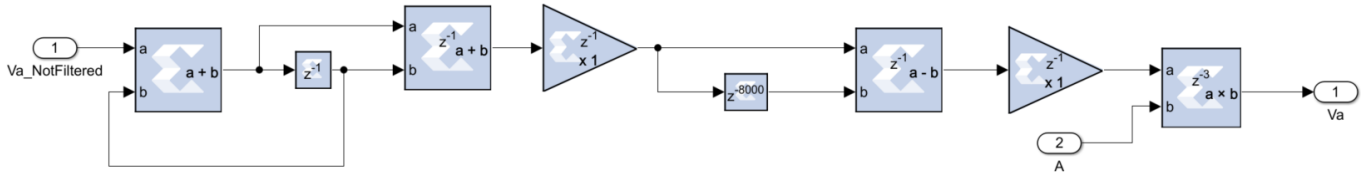


Figure 63 Mean Value Filter (screen from Simulink)

The three-phase voltages output from the inverter have high frequencies and therefore before being exploitable by the motor (implemented on the processor) must be re-transformed in reference systems d-q through the direct transforms of Clarke & Park. In doing so, the problems related to high dynamics are overcome.
To get an idea, the cycle has reached the point indicated in the figure:

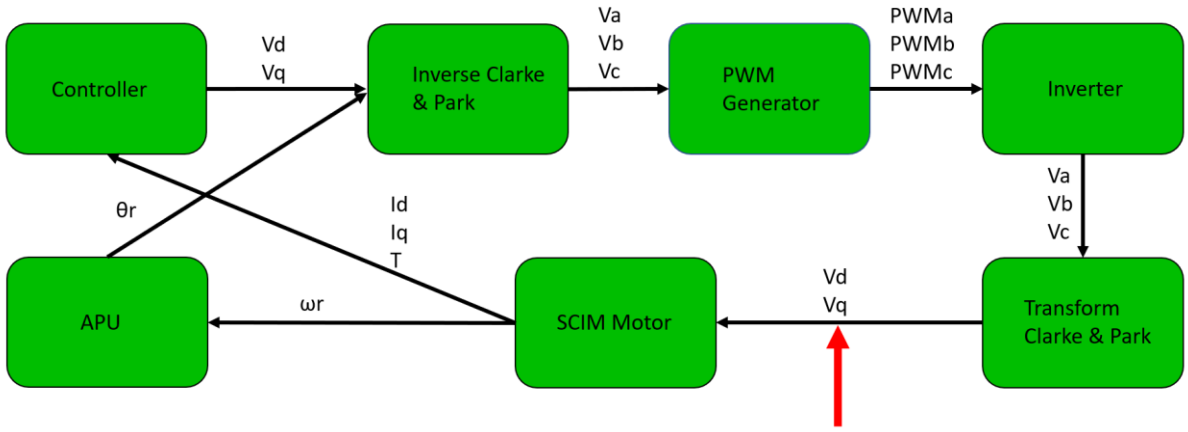


Figure 64 Global system overview

As you can see at this point the loop is definitely closed, as it has returned to the point of start of the modeling, that is to the motor.
Having learned all the modelling and understood the individual blocks better, it is now perhaps easier to give a final reading of the whole principle of operation of the project in its entirety.
The asynchronous motor turns at a certain speed generating a certain current. These quantities are taken and analyzed by a control unit, which acquires these data and compares them with a target speed decided by the user. If the data read match the desired data, then the ECU does not intervene, otherwise it applies the corrective action necessary so that the future speed reached by the engine is exactly the same as the user's request. Output from the controller then there is a sinusoidal voltage that must be set to the motor, which will respond by accelerating or slowing down. Intuitively, if its current speed is too high then the voltage it receives will be lower than the one it received previously, otherwise it will see a voltage increase in its windings.

However, before reaching the motor the three-phase voltage is used to power the inverter, which is necessary to make AC the battery voltage of the vehicle, which would otherwise be in DC.
The last mage of this chapter summarizes which elements are on the processor and which one are on the FPGA.

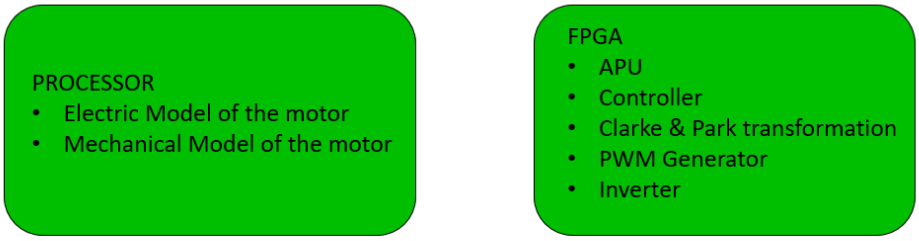


Figure 65 Recap of the components

10. SOFTWARE PROCEDURE

Defined how the system was modeled now the focus is shifted to how you proceeded to run the model in a real time test-bench. The procedure includes several steps that accompanied all the thesis activity and all the versions of the motor-inverter model that have been created.

The first step was the offline simulation on Simulink. This step was the simplest and most natural. Once the libraries were installed in order to use the FPGA blocks, the modeling proceeded smoothly. Among the blocks that are definitely worth remembering is the pair System Generator and FPGA Setup. The first is necessary to be able to generate a file that is simulable, while the second contains all the parameters of the FPGA card that you want to simulate.

Figure 66 System Generator and FPGA Setup (screen from Simulink)

Here can be noticed the big difference between the capability of simulation on the processor (offline period) and how fast is the FPGA, which can operate up to the ns. If the processor step simulation can be freely tunable, the on line's one depends on the FPGA itself and it is a fixed value.

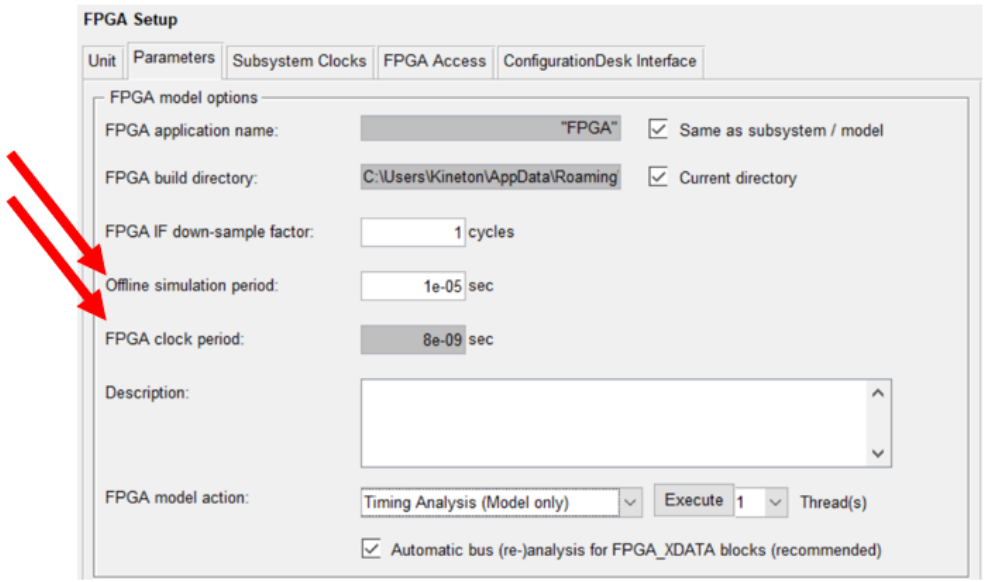


Figure 67 FPGA specific parameters (screen from Simulink)

Downstream of this introduction on how to implement external library blocks on simulink it is time to explain in detail how it was possible to test on a real time simulator the model just described.
Now will be explained all the steps of the road block that allowed to get to the final goal, that is to have a type of executable file on 'HIL.

TIMING ANALYSIS

By definition, each block of the XSG library is characterized by a latency. This parameter was fundamental for all the components that were modeled on FPGA. The latency expresses at the same time both the time necessary to complete a precise operation and the maximum time available in which to complete it. More complex operations require higher latency: the calculation of the value of the sine given an angle is clearly more challenging to compute than a bit-to-bit comparison in a logical operation or a multiplication. This explains the different latencies indicated in the following figure.

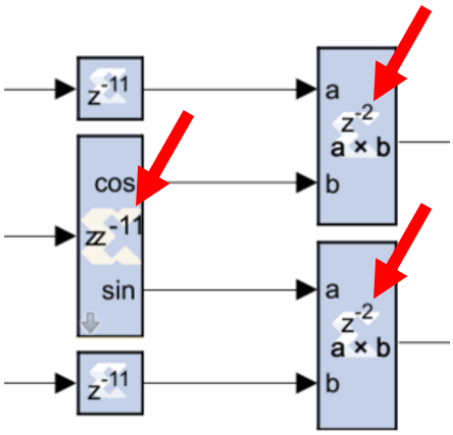


Figure 68 Latency of the blocks (screen from Simulink)

A different argument is made for delays: these square blocks are simply used to synchronize the inputs so that the n-th bit of an input, corresponding to the moment t0, is actually computed with the n-th bit of a second magnitude also corresponding to time t0. In the photo in question, to ensure that all the inputs coming from the outside and the one calculated as a result of the sine were multiplied at the same time bit to bit.

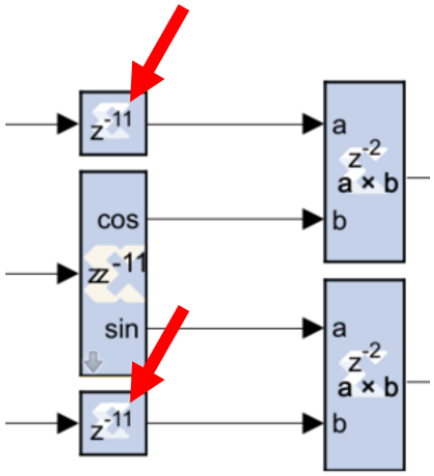
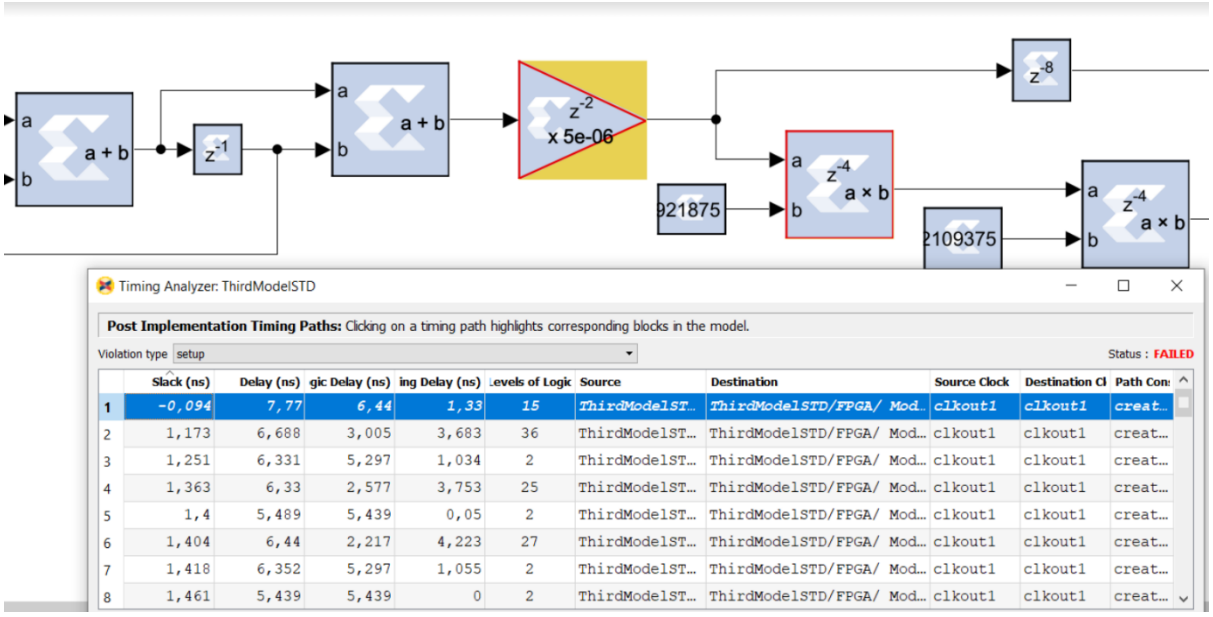


Figure 69 Delay to synchrnoize the inputs (screen from Simulink)

It is the same image but with the focus placed on different blocks. The first expresses the time necessary for an operation to end, the second emphasizes the need to synchronize the inputs. The need to impose a certain latency has complicated the modeling, as downstream everything has always been performed so-called timing analysis. This (long) process consists in verifying that all successive operations of the FPGA are actually completed within the prescribed time frame.

Often and willingly, we had to repeat several times a timing due to blocks of which we underestimated the time needed to complete all the calculations. At the end of the timing analysis in fact the solver always summarizes the state of the analysis, reporting the following result in case of a failed attempt



	Slack (ns)	Delay (ns)	gic Delay (ns)	ing Delay (ns)	Levels of Logic	Source	Destination	Source Clock	Destination Cl	Path Con
1	-0,094	7,77	6,44	1,33	15	ThirdModelST…	ThirdModelSTD/FPGA/ Mod…	clkout1	clkout1	creat…
2	1,173	6,688	3,005	3,683	36	ThirdModelST…	ThirdModelSTD/FPGA/ Mod…	clkout1	clkout1	creat…
3	1,251	6,331	5,297	1,034	2	ThirdModelST…	ThirdModelSTD/FPGA/ Mod…	clkout1	clkout1	creat…
4	1,363	6,33	2,577	3,753	25	ThirdModelST…	ThirdModelSTD/FPGA/ Mod…	clkout1	clkout1	creat…
5	1,4	5,489	5,439	0,05	2	ThirdModelST…	ThirdModelSTD/FPGA/ Mod…	clkout1	clkout1	creat…
6	1,404	6,44	2,217	4,223	27	ThirdModelST…	ThirdModelSTD/FPGA/ Mod…	clkout1	clkout1	creat…
7	1,418	6,352	5,297	1,055	2	ThirdModelST…	ThirdModelSTD/FPGA/ Mod…	clkout1	clkout1	creat…
8	1,461	5,439	5,439	0	2	ThirdModelST…	ThirdModelSTD/FPGA/ Mod…	clkout1	clkout1	creat…

Figure 70 Failed timing analysis (screen from Simulink)

As you can see, the report shows which block led to the failure of the analysis. To solve the problem, it is necessary to increase the latency assigned to the blocks in its surroundings in order to allow the operations to be properly performed. This is the state of a timing performed successfully.

```
Timing analysis finished.
Elapsed time is 00:56:06.
```

Timing Analyzer: ThirdModelSTD

Post Implementation Timing Paths: Clicking on a timing path highlights corresponding blocks in the model.

Violation type: setup — Status : PASSED

	Slack (ns)	Delay (ns)	gic Delay (ns)	ing Delay (ns)	Levels of Logic	Source	Destination	Source Clock	Destination Cl	Path Constraints
1	1,462	6,215	1,968	4,247	21	ThirdMod...	ThirdMod...	clk	clk	create_clock -name clk -...
2	1,504	5,439	5,439	0	2	ThirdMod...	ThirdMod...	clk	clk	create_clock -name clk -...
3	1,504	5,439	5,439	0	2	ThirdMod...	ThirdMod...	clk	clk	create_clock -name clk -...
4	1,504	5,439	5,439	0	2	ThirdMod...	ThirdMod...	clk	clk	create_clock -name clk -...
5	1,504	5,439	5,439	0	2	ThirdMod...	ThirdMod...	clk	clk	create_clock -name clk -...
6	1,504	5,439	5,439	0	2	ThirdMod...	ThirdMod...	clk	clk	create_clock -name clk -...
7	1,504	5,439	5,439	0	2	ThirdMod...	ThirdMod...	clk	clk	create_clock -name clk -...
8	1,504	5,439	5,439	0	2	ThirdMod...	ThirdMod...	clk	clk	create_clock -name clk -...
9	1,504	5,439	5,439	0	2	ThirdMod...	ThirdMod...	clk	clk	create_clock -name clk -...
10	1,504	5,439	5,439	0	2	ThirdMod...	ThirdMod...	clk	clk	create_clock -name clk -...
11	1,504	5,439	5,439	0	2	ThirdMod...	ThirdMod...	clk	clk	create_clock -name clk -...
12	1,504	5,439	5,439	0	2	ThirdMod...	ThirdMod...	clk	clk	create_clock -name clk -...

OK Help

Figure 71 Passed timing analysis (screen from MATLAB)

Intuitively, the higher the delay value, the more likely the timing analysis is to be successful. This is obviously true, but you have to collide with the heaviness of the model and the time required to do so. With the more complex models it has also come to perform timing for more than an hour. From here we can understand the need to find a right compromise between the possibility of passing the timing analysis, and then put high latencies, and the duration of operations.

BUILD PROCESS

Once the timing analysis is complete the next step is to start creating a file type that can run on configuration desk in view then to have the final executable to run on the HIL.

To summarize, at the time of timing analysis you are still working on a file .slx. The build process is the operation in which the Simulink model is converted to VHDL code. This is a very complex and computationally heavy operation, requiring not to use the PC at the same time as the build itself. The word "DO NOT EDIT" that appears during this process says a lot about it. The longest builds lasted an hour and a half. Here is the screen of what returns the MATLAB command windows at the end of the process.

Type	Used	Available	Utilization [%]
Configurable Logic Block Slices (LUTs, Flip-Flops)	6509	25350	25.68
Configurable Logic Block Slice LUTs	11744	101400	11.58
Configurable Logic Block Slice Flip-Flops	17827	202800	8.79
Block RAM Blocks 36 Kb	8	325	2.46
Block RAM Blocks 18 Kb	12	650	1.85
DSP Slices	24	600	4.00

```
FPGA Build Done
Elapsed time is 00:25:54.
```

Figure 72 Build process (screen from MATLAB)

It is very important to remember that the creation of the VHDL is specific to the type of FPGA you want to configure, hence the need to set in FPGA Setup the correct values of the card used in the HIL (DS2655M1).

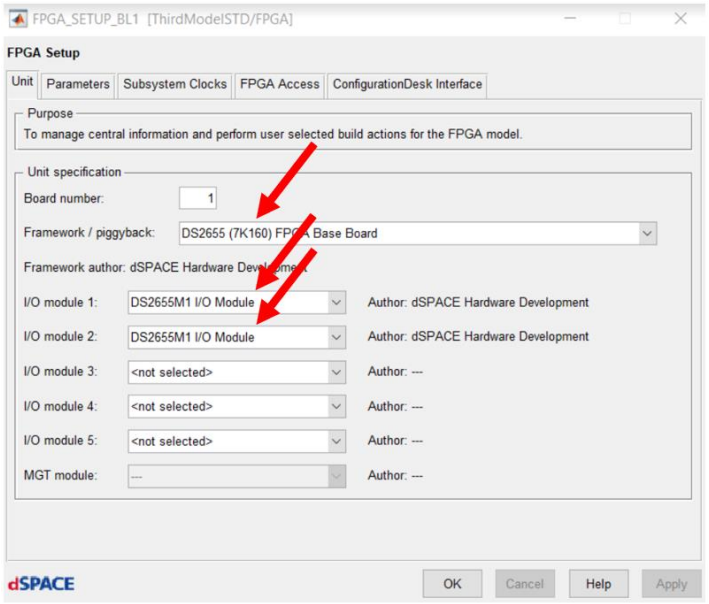


Figure 73 Setting the hardware needed for the configuration (screen from Simulink)

The output of these operations is a file with extension .ini.

CONFIGURATION DESK

Switching to configuration desk is made possible thanks to the "Model Separation Setup" block. That said, the interface with the model configuration was a step consisting of several steps.

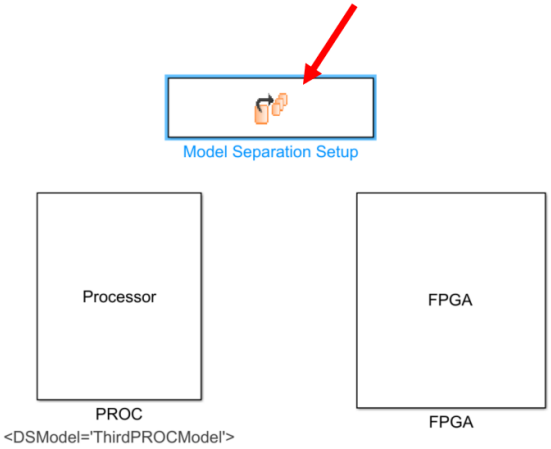


Figure 74 Model Separation Setup (screen from Simulink)

The following image is very important because underlines the relationship between Simulink and Configuration Desk:

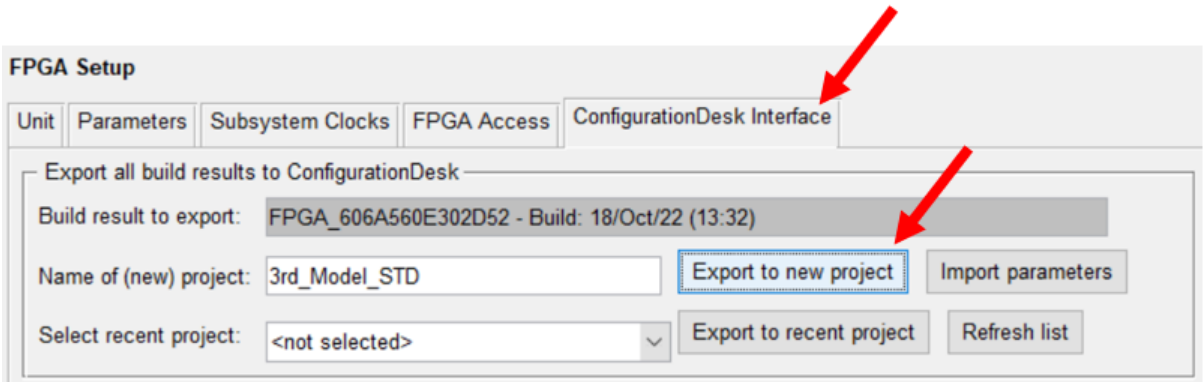


Figure 75 Setting the export from Simulink to Configuration Desk

The first line is the result of the build. Starting from this result you can name a new project, which is linked to the .ini, and export the project to prepare the configuration.

In general, configuring a model means setting up the physical I/O channels of the FPGA so that the simulated quantities inside of the HIL can be brought outside the simulator. Not surprisingly, when generating the final configuration desk file, it is necessary to describe the 'hardware on which the model is then dropped. It is a. htfx file that describes the architecture used and is unique to each FPGA card.

In the following image you can see the possibility to import the hardware of the HIL, while on the left there is what' inside the .htfx, which are the hardware components mounted inside the HIL.
This is a crucial part of the software process because to the VHDL language expressing the model is associated a series of physical channels.

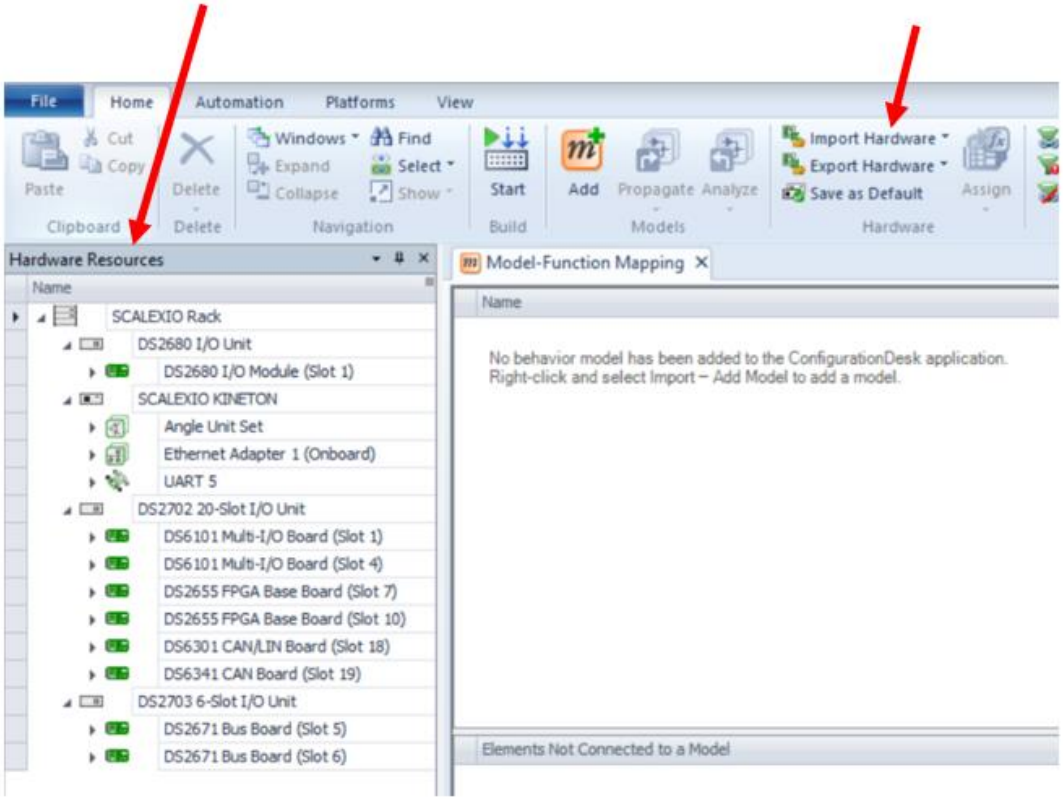


Figure 76 Configuration of the hardware of the HIL (screen from Configuration Desk)

To be clear, the xx channel of a given card corresponds to a DigIn (for example), while on a yy card it could correspond to an AnalogOut. The final result generated by this second build is an .sdf file that can finally be run on the Control Desk in the HIL test bench.

CONTROL DESK

The last step in the software procedure is control desk. This simulation tool is widely used in the automotive sector, as it allows to simulate any vehicle function in order to recreate the necessary scenarios required during the verification and validation phase.
More specifically, if the model being simulated is subsequently accurate, it is possible to see the messages exchanged between the different CAN control units, to monitor the behavior of the vehicle battery, to impose a certain dynamic to the vehicle through the pedals and the gearbox, check with the battery in power latching, and so on.
Control Desk allows you to view the trend of the variables as the system evolves: in the interface that appears when you run the model there is a tab, called variables, from which you can select quantities to be monitored in real time by means of oscilloscopes, vectors and other tools required for testing.

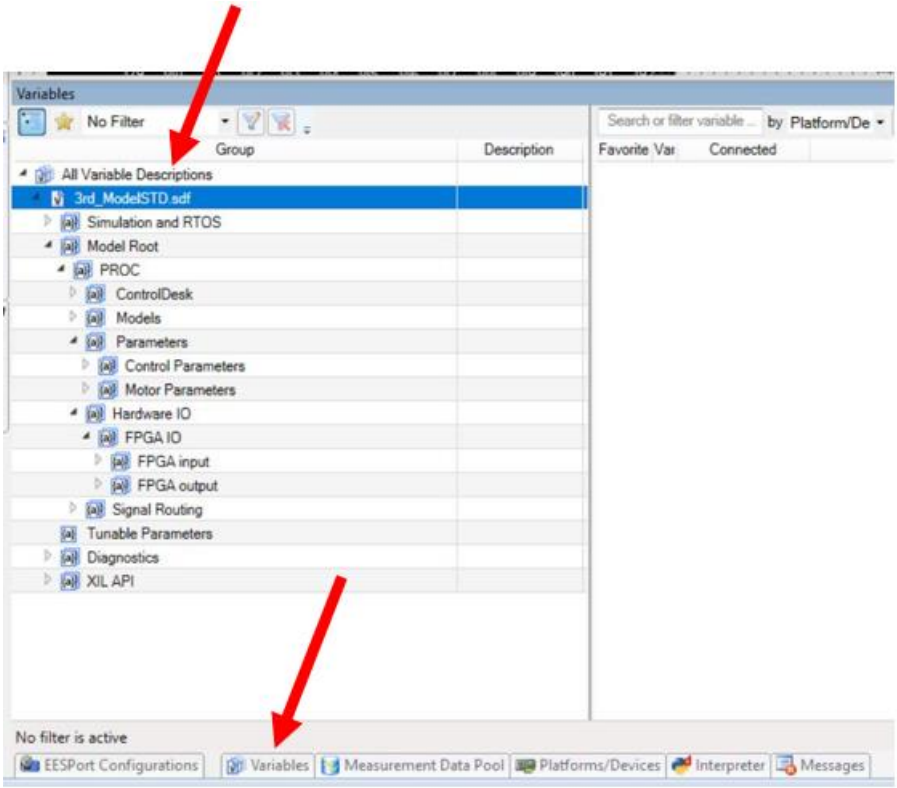


Figure 77 Variables panels (screen from Control Desk)

In order to start the simulation and test the behavior of the control unit you need to "go online" with the project. In a nutshell, going online means connecting the entire architecture, starting from the simulated model on the processor, to the HIL with all its peripherals and the DUT that is being tested.
Once all the elements are properly connected you can run the simulation.

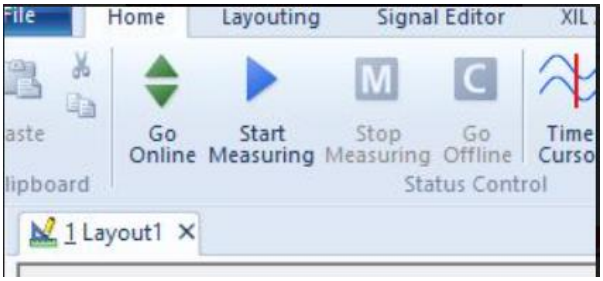


Figure 78 Go Online e Start Measuring buttons (screen from Control Desk)

Once all the acquisitions are finished, before you can close the whole project and "turn off" the HIL you have to do the reverse procedure, then go offline so disconnect all the components.

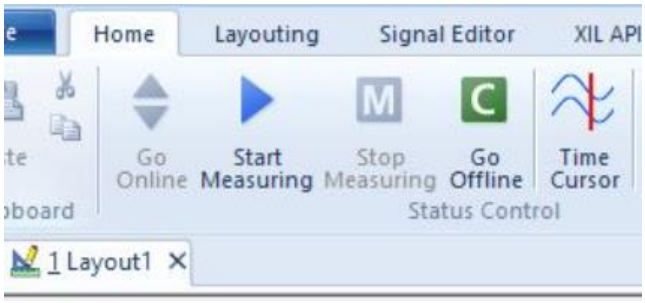


Figure 79 Go Offline button (screen from Control Desk)

Only after going offline you can unload the file .sdf loaded on the HIL. This last operation is managed by a special tab in the Control Desk menu.

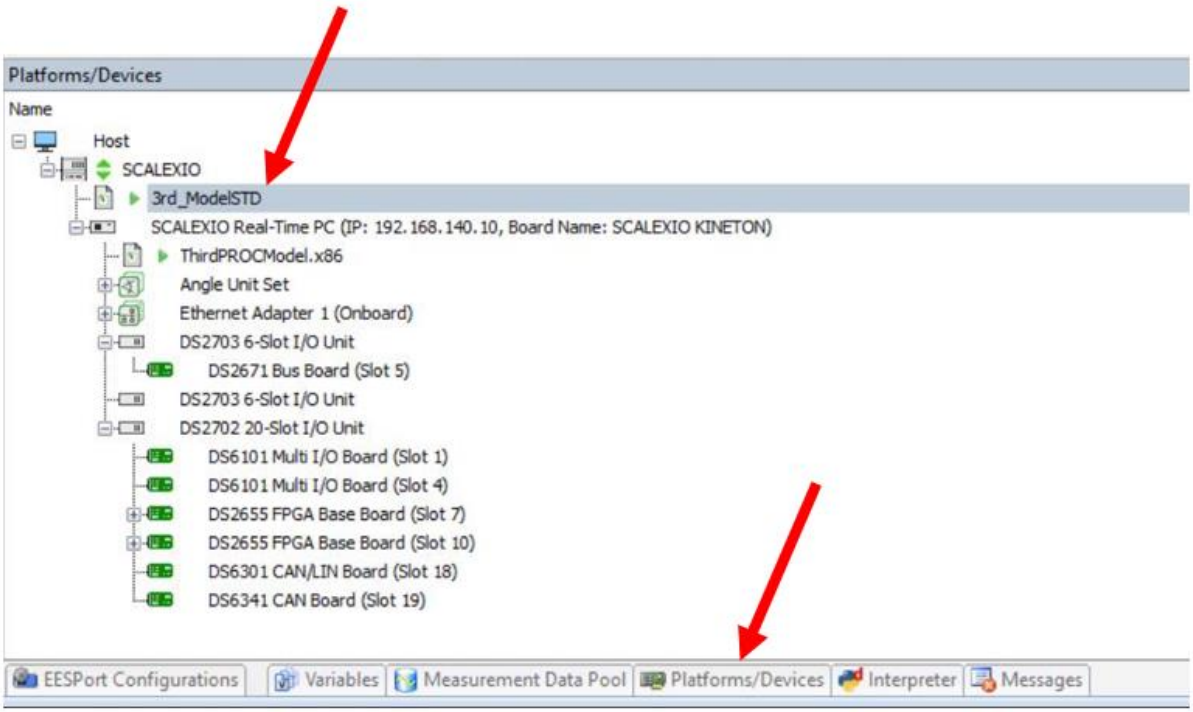


Figure 80 Platform/Devices to unload the .sdf (screen from Control Desk)

The next image briefly summarizes all the software procedure needed to get an executable file on the simulator from a Simulink model.

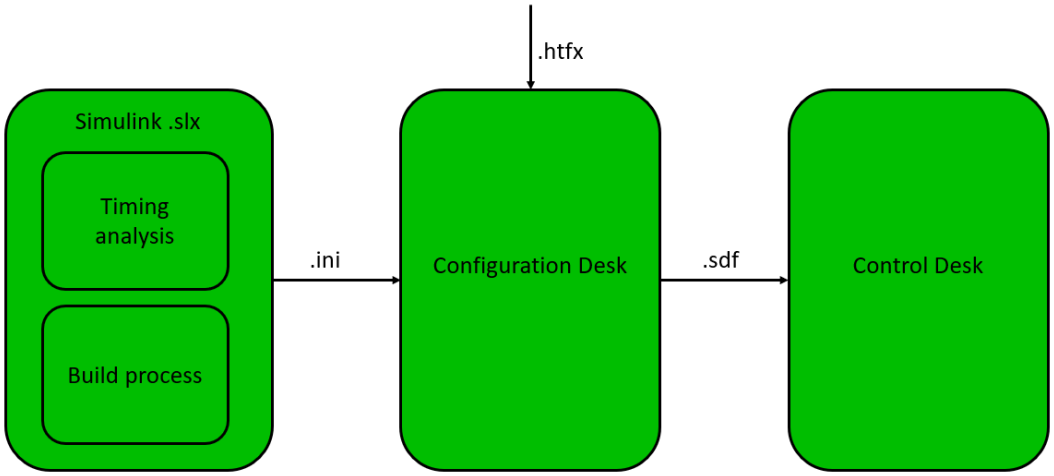


Figure 81 Recap of the complete software procedure

11. RESULTS AND CONCLUSIONS

Finally, the discussion has reached the final stages. At this point, note the modeling of the motor-inverter model in all its parts, and known as this was dropped on the simulator, it's time to collect the results obtained from the acquisitions.

The first consideration to make is transversal to all the versions that have been tested, both on HIL and on the processor: the model is to be considered working when it is able to follow the speed set by the controller. This statement is obvious but it was the parameter by which the functionality of the model was evaluated. The more the output follows the input, the more accurate the system response.

One of the biggest difficulties encountered during the modeling was to configure a motor that was parametric, therefore able to operate with any external control unit. In other words, it was difficult to create an engine that was robust enough to work, ideally, with any value that has a similar meaning. It's logical that the model won't work if the supply voltage is set to millions of volts, and so on. Below is the list of parameters used and the formulas to calculate them.

```
%% Mechanical Parameters

J  = 0.5;                    % Moment of inertia [kgm^2]
P  = 2;                      % Pairs of Pole
F = 0;
Tl = 0.05;                   % Load Torque [Nm]
Pn = 160000;                 % Nominal Power [W]
sn = 0.01;                   % Nominal Slip
```

Figure 82 Mechanical parameters (screen from MATLAB)

```
%% Electical Parameters

Vdc = 1287;                                % Nominal Voltage
fn = 84;                                   % Frequency [Hz]
Rr = 0.103;                                % Rotor resistance [ohm]
Rs = 0.223;                                % Stator resistance [ohm]
Lm = 0.0438;                               % Magnetizing inductance [Hr]
L1s = 0.00158;
L1r = 0.002076;
Lr = Lm+L1s;                               % Rotor inductance [Hr]
Ls = Lm+L1s;                               % Stator inductance [Hr]
sigma = 1-Lm^2/(Ls*Lr);                    % Sigma
w = 2*pi*fn*P;                             % Nominal Syncronus Speed [rad/s]
ns = 60*fn/P;                              % Nominal Syncronus Speed [rpm]
Zeq =(Rs+i*w*L1s)+(i*w*Lm*(Rr/sn+i*w*L1r))/(i*w*Lm+(Rr/sn+i*w*L1r)); % Impedence
Is= Vdc/Zeq/sqrt(2);                       % Is
Id_ref = real(Is);                         % D-axis Reference Current
Iq_ref = abs(imag(Is));                    % Q-axis Reference Current
Iq_max = 300;
```

Figure 83 Electrical parameters (screen from MATLAB)

```
%% Controller Parameters

Kp_speed = 50;
Ki_speed = 500;

Kp_i = 1;
Ki_i = 100;

T_sample =  8e-9;
FPGA_Step = 8e-9;
integral_step = 8e-9;
T_PWM = 8e-5;
Fin = 1/T_PWM;

Den = 1/(P*Lm^2*Id_ref);
```

Figure 84 Controller parameters (screen from MATLAB)

```
%% Inverter Parameters
T = 1/(fn);
RC = 2*T_PWM;
```

Figure 85 Inverter parameters (screen from MATLAB)

The difficulty is linked to the fact that the real control units and the real motor models are characterized by many controls in order to manage values out of range (maybe linked to electrical fault on time) for example, that in the case at hand were not considered. The difference is that in the model here treated, values out range have always led to a completely wrong result by overflowing the model.
In the possible improvements at the end of the work will be reported the models of the dSPACE to understand the complexity related to the need to manage any input value without generating an output that explodes.

First of all a premise, the first broom at the top depicts the ideal value that the model should assume. The second scope follows the output variable of the ω_r model. As you can see the output of the model follows exactly the set calibration speed (except for a certain delay). For completeness of results, it was tried to increase the speed and then to reduce it, so as to be sure that everything worked correctly. Knowing what is behind the control algorithm, one thinks that in the first part of the simulation the engine is supplied with more and more current in order to reach the nominal speed. From the 13th second instead, the trend is reversed: when the engine is required a sharp slowdown then the current is quickly scaled to return to converge on the ideal value.
Finally, the last table shows the values of the controller parameters, which can be changed as desired.

In both images, of course, the parameters were kept constant so that you can check the correct functioning of the model in all possible working conditions. For now, the attention is more focused on the output result rather than on the fact that it is parametric.

Below are the results obtained directly from the Control Desk.

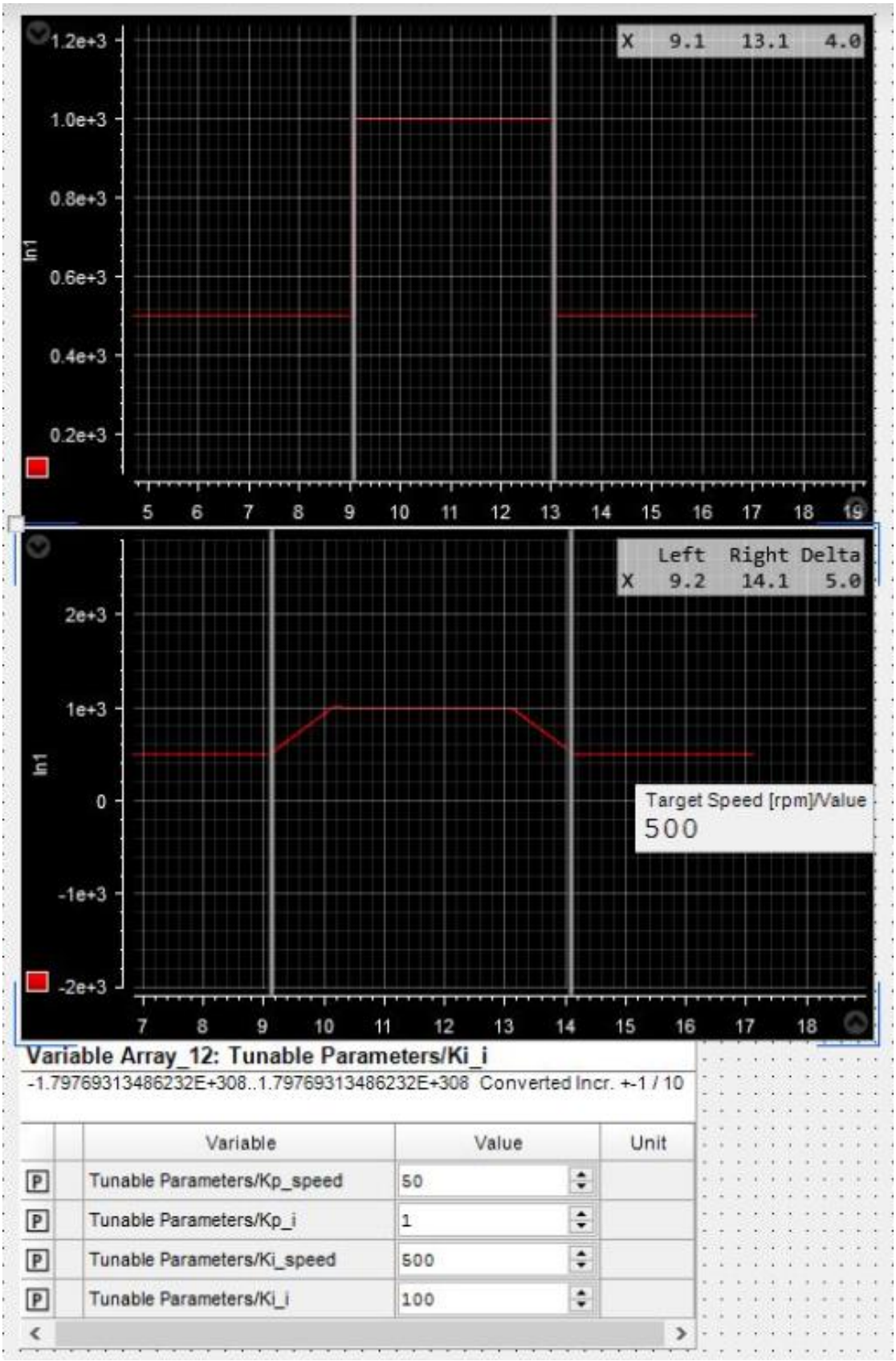


Figure 86 Desired vs actual behavior: rotor speed (screen from Control Desk)

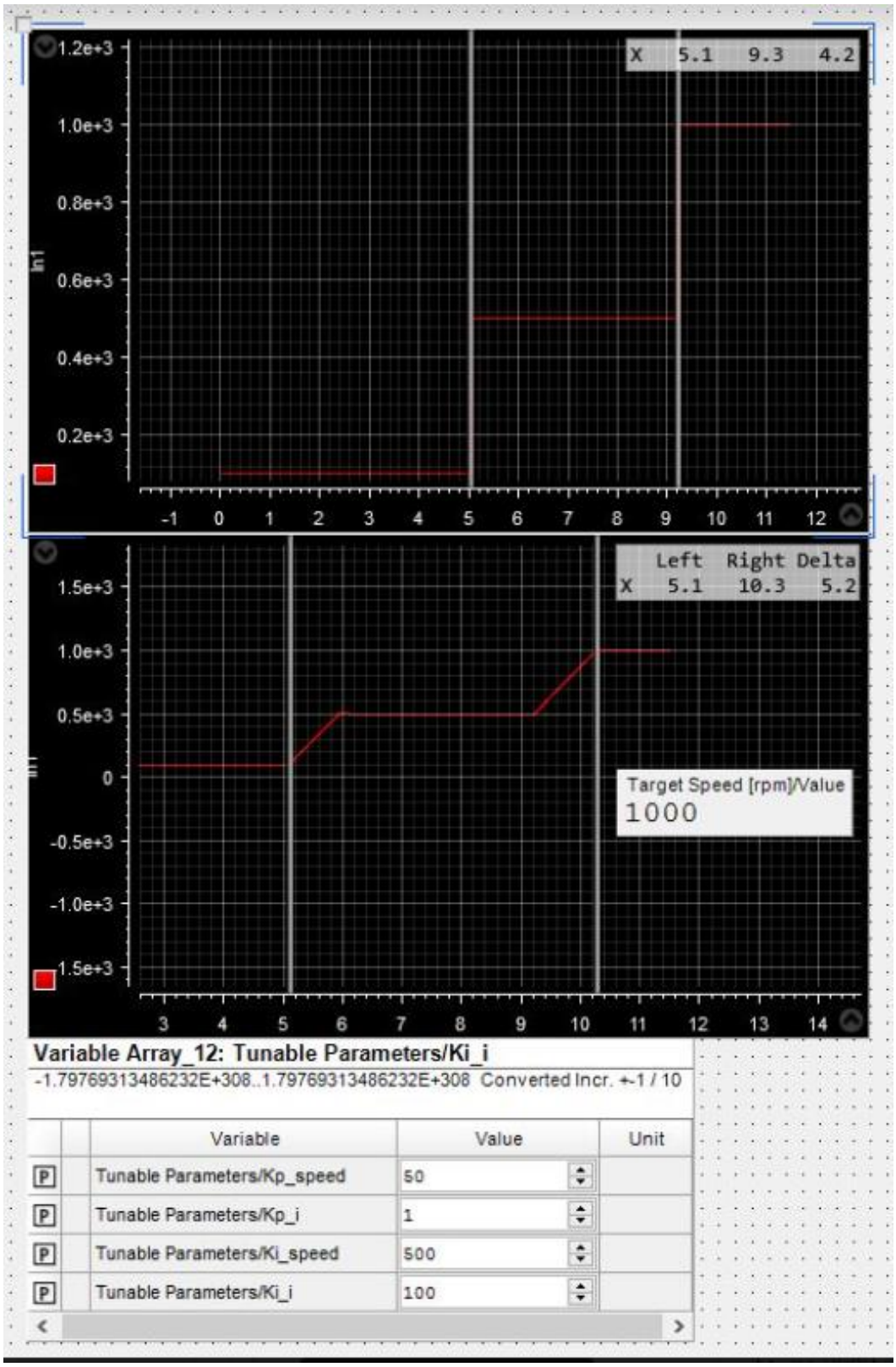


Figure 87 Desired vs actual behavior: rotor speed (screen from Control Desk)

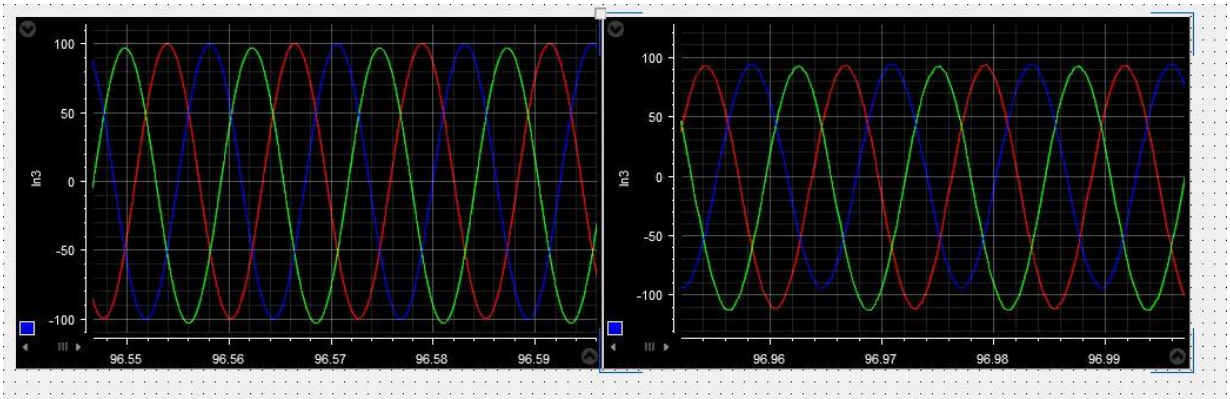


Figure 88 Desired vs actual behavior: inverter voltage (screen from Control Desk)

Once the functionality of the model has been verified, we now report the differences obtained by setting slightly different parameters.

For convenience, below is a table that reads the two classes of parameters so that later you can refer to class A and class B in a unique way.

Class A:

Kp_speed	50
Kp_i	1
Ki_speed	500
Ki_i	100

Table 1 Class A parameters

Class B:

Kp_speed	70
Kp_i	7
Ki_speed	700
Ki_i	700

Table 2 Class B parameters

First of all let's talk about overshoot:

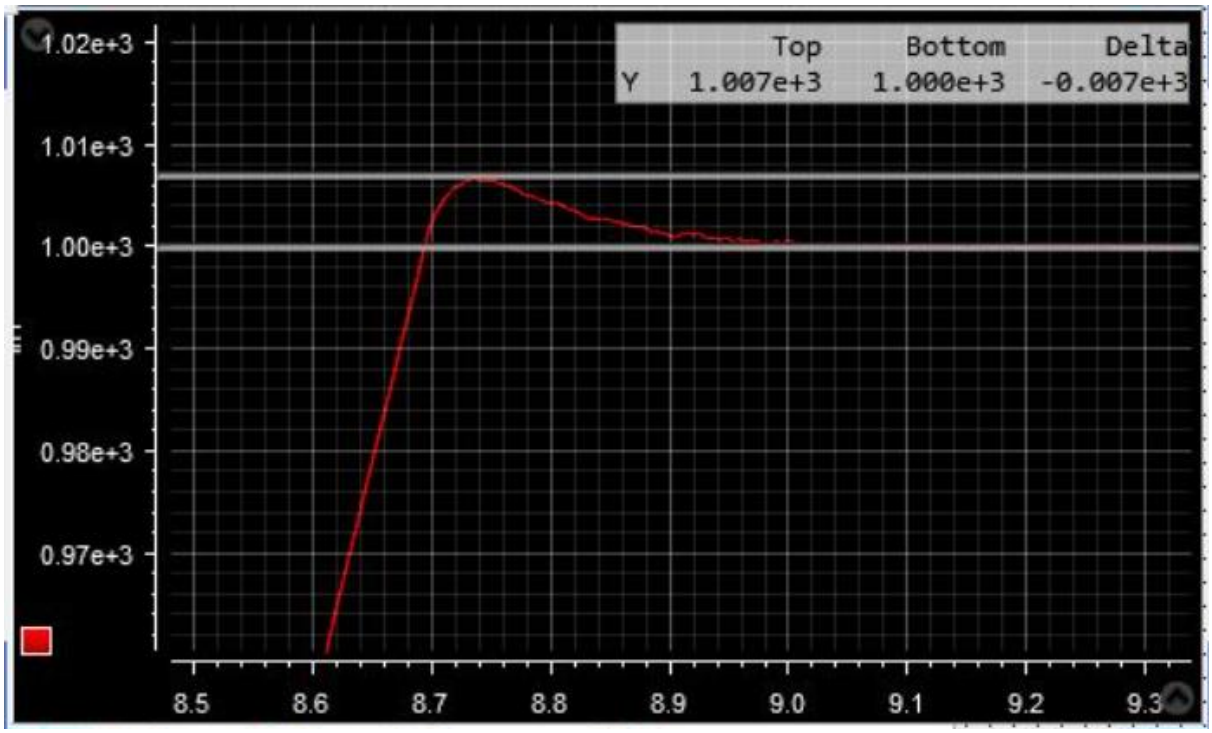


Figure 89 Overshoot with class A parameters (screen from Control Desk)

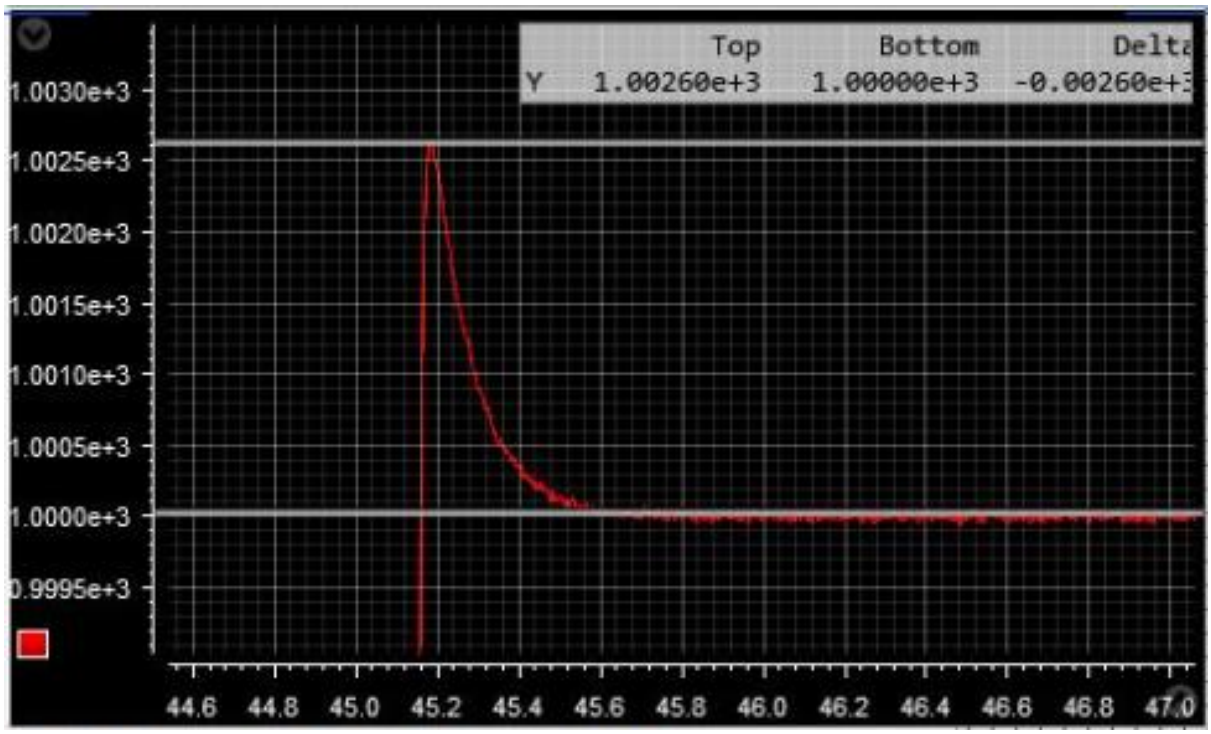


Figure 90 Overshoot with class B parameters (screen from Control Desk)

As we can clearly see, changing the value of the parameters a little, we have very different behaviors:

$$\hat{y}_A = 0.007$$
$$\hat{y}_B = 0.0026$$

Without anticipating the reasons for this difference is also attached another screen. This aims to highlight another difference, namely the time required to reach the steady state:

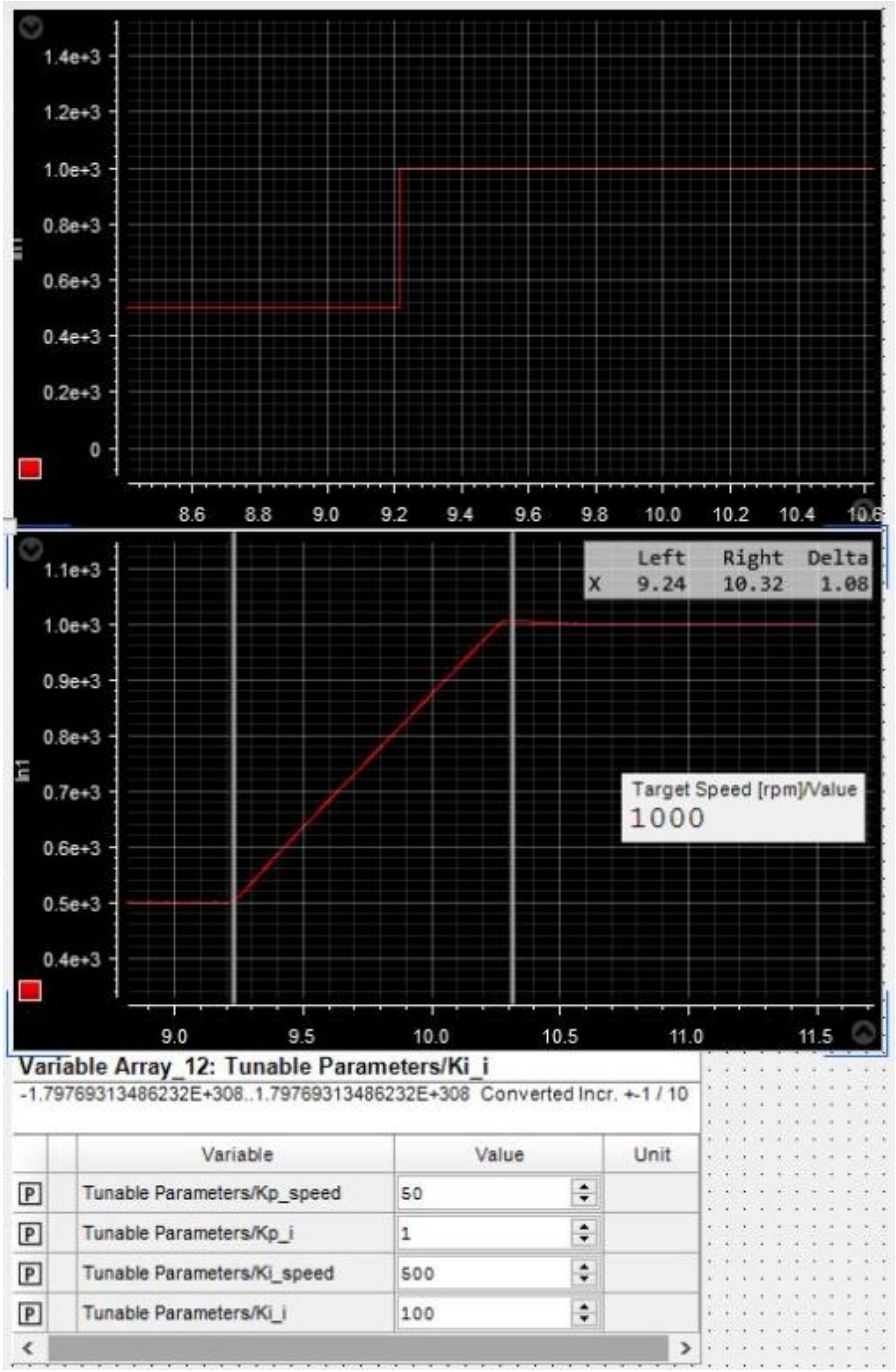


Figure 91 Desired vs actual behavior (screen from Control Desk)

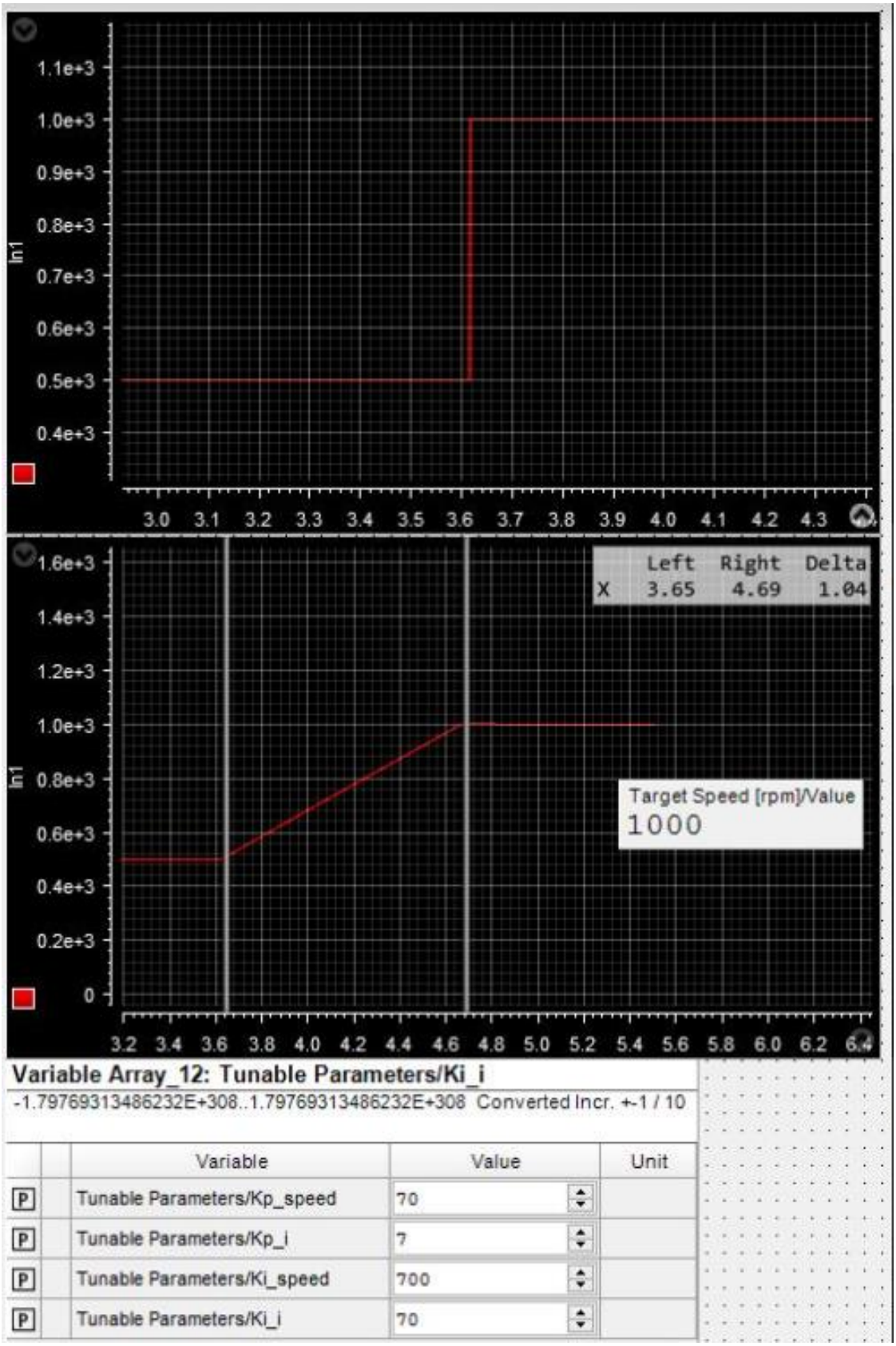


Figure 92 Desired vs actual behavior (screen from Control Desk)

As anticipated, this time the focus is on the time difference, delta, needed to reach the target speed.

$$\Delta_A = 1.08$$
$$\Delta_B = 1.04$$

As is easy to guess, these two differences are related to the parameters of the model. A few pages before we talked about the difficult balance behind the tuning of the characteristic parameters of the controller. In general, a faster control produces a greater overshoot. Conversely, to have a more linear control and behavior, more time is required. This speech is general, but as we can see in the specific case, the class B seems better as it guarantees a faster response with a minor overshoot.

All the screens just presented are taken from the Control Desk oscilloscope.
Below are some screens taken from a real oscilloscope. First some images are reported of the wiring harness and a screen of the Hypertac mask to understand for what each pin stays for:

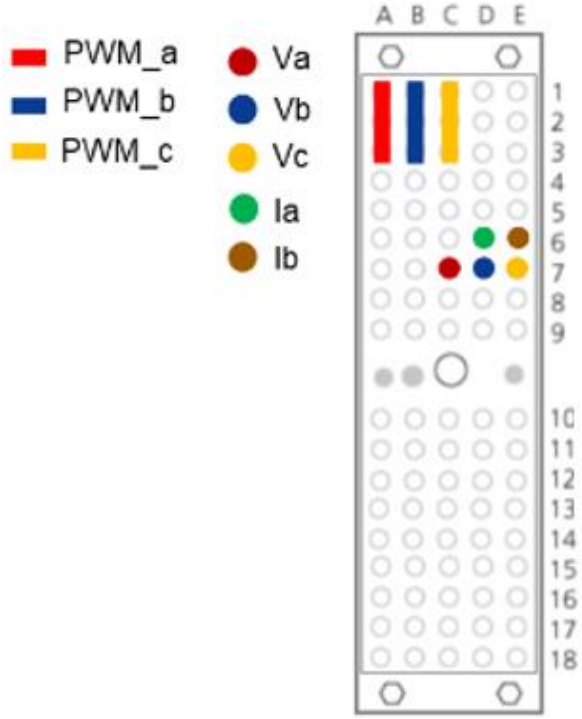


Figure 93 Hypertac port

Starting from this basic knowledge we can guess how the next photo foresees a wiring in order to manage PWM. In particular this step is important as the wires enter and exit the HIL, so the signal does the same.
This step demonstrates the possibility of connecting an external control unit to the configured model. In the thesis in question the "jumpers" with digital signals enter and exit from the HIL,

but in reality those cables are replaced directly by those of the DUT since the pins are already configured to manage digital signals in and out

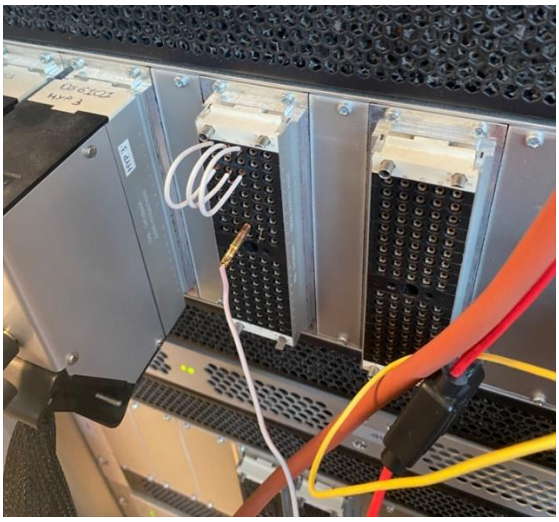


Figure 94 Harness for the PWM

The next image is very significant as it allows you to see the real application that for now has always been just explained. The big pc in the background is the HIL with all its wiring. On the right you can see the power supply, then you recognize the black cable bundle that passes the signals to the fault injection unit next to the oscilloscope. Finally in the foreground is the oscilloscope. This is connected to the FPGA by the two red and yellow probes. The sine waves that are depicted are the (two of three) three-phase voltages that come out of the inverter.

Figure 95 Wiring harness

The next images that will be presented are the real central point of the thesis as they summarize the true potential of 'HIL dropped within the software verification.

Explained one last time, the concept of 'HIL allows you to test a real physical component simulating everything around him, making him believe to work in a normal working condition. This type of simulation is possible because the DUT receives real signals from the HIL channels that are processed as if they came from other adjacent components. In the same way the outputs produced are sent back to the HIL through the input channels. In this way the tested component behaves as if it were really part of a chain as it receives signals from what precedes it and produces outputs for what follows it.
Starting from these considerations, the next images represent the analog (in the case of sinusoids) and digital (in the case of PWM) signals that are output from the model.

The first represents the output sinusoidal voltages from the inverter and are then sent to the motor. These waves are the voltages that are controlled by the controller in order to obtain the desired speed. As you can see, they are rather clean sinusoidal. They are the Vabc that are often referred to in the mathematical treatment of the motor model. Unfortunately, on the oscilloscope it was possible to visualize only two waves at the same time and not all three because the oscilloscope is characterized by the presence of only two probes.

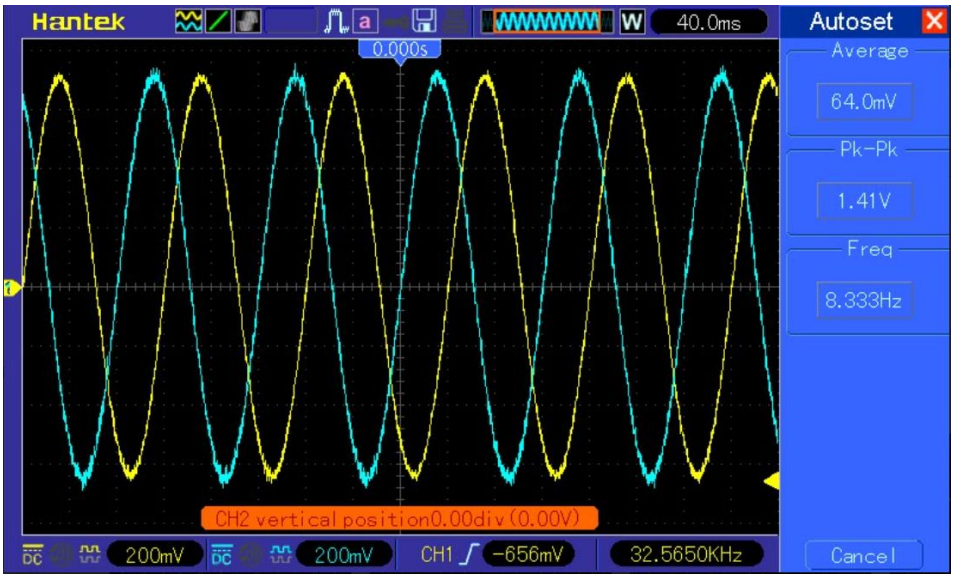


Figure 96 Analog voltage exiting from the inverter (screen from the oscilloscope)

The other image below is the PWM generated by the PWM generator. These are obtained based on the comparison of the reference sine wave output from the controller with a triangular wave characterized by a high frequency.

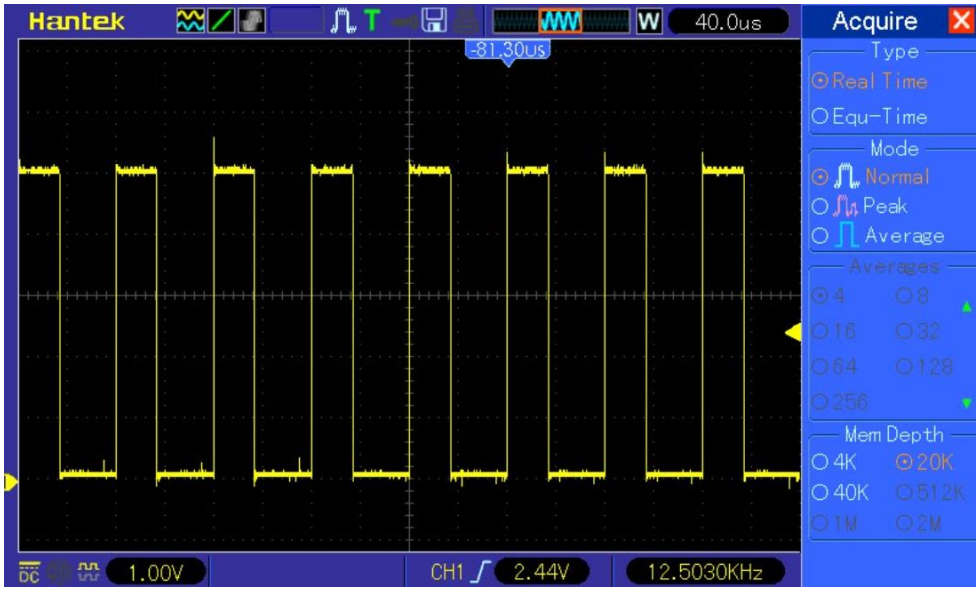


Figure 97 PWM entering inside the inverter (screen from the oscilloscope)

Finally, we report a screen of the results obtained by changing the parameters of the model. As you can see, the output-controlled variables are completely wrong and without any physical meaning. This seemingly wrong result is actually related to the parameters selected.

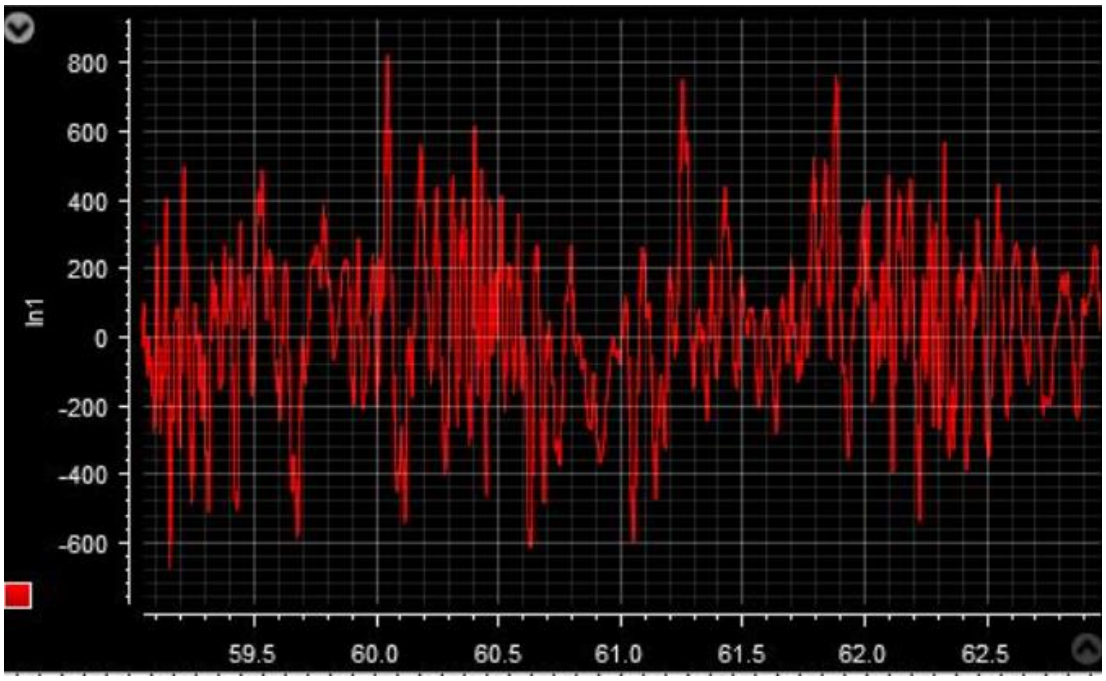


Figure 98 Wrong output (screen from Control Desk)

This result brings two important considerations: the first is linked to the possibility of using the model with different sets of parameters and therefore ensures the possibility of testing different control units would be very useful. This fact if applied in the real automotive industry

translates into a huge decrease in costs and development hours as with a parametric model only it is possible to test different units. On the other side the project just presented is not very robust when passing wrong variables. From the point of view of the model this is simply linked to the choice of incorrect values, but from an application point of view it could happen that the parameters are correct but that there is a malfunction in the model, such as electrical failures. This circumstance is usual in the automotive world, so it is important to have safety controls to cope with these eventualities that can also be very dangerous for safety.
This final necessity leads the discussion of the thesis directly to the proposals for improvement, which include additional controls in order to stem any possible spikes in the values of the variables.

12. POSSIBLE FUTURE DEVELOPMENTS

In this last part of the report, we want to stress a series of changes that could lead to an improvement in the performance of the system in terms of speed achieved, in terms of speed of response and stability in handling values out of range.
To tell the truth you could make a lot of changes in order to make the model more similar to reality, such as to simulate the operation of a 12V battery with all its component. An additional change at the application level could be to insert (or model) a transmission of power, a gearbox, this would allow to work the engine at an always optimal number of revolutions, this ensures lower battery consumption and greater efficiency. Such a solution could bring benefits in terms of application, but in the model nothing would change in terms of performance.
Here we will focus more on the changes inherent in the parts modeled in this thesis.
The question of finding greater robustness in response to parameter changes does not concern the engine model as much as the controller. Before going into this the first consideration, trivial, that can be made is undoubtedly to implement the engine model on FPGA. It has been said several times that the big difference between processor and FPGA lies in the speed of calculation, therefore as the speed increases it is logical that having all the model able to process the data with a frequency of ns and not of ms would improve the accuracy of the calculations (avoiding any problems of aliasing).

The latest improvement proposals are undoubtedly related to the components that have been most problematic throughout the modeling, namely motor controller and inverter.
To understand the real modeling complexity of these components, we report screens of the corresponding models made by the dSPACE.
There are two improvements that can be made in the inverter: the first is linked to a better physical likelihood. In the thesis in question the inverter was modeled using an RC filter, although in reality the motor is an inductive load. For this reason, a first improvement could be linked to the implementation of the equations of an RL filter.
The second improvement is functional: in the first pages of the thesis, it was said that the inverter is composed of 6 transistors in parallel that generate sinusoidal waves. The choice not to implement this solution is linked to the complexity of the nonlinear equations that characterize it.
Below is a screen but that will not be explained in detail because of the complexity. For a complete understanding we refer to the site of the dSPACE.

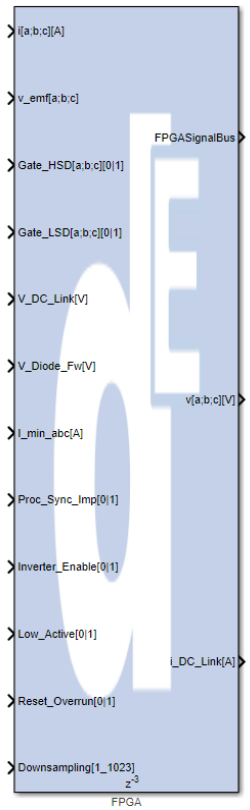


Figure 99 Inverter dSPACE (screen from Simulink)

The next image is the inside of the above block. Here it is for illustrative purposes only

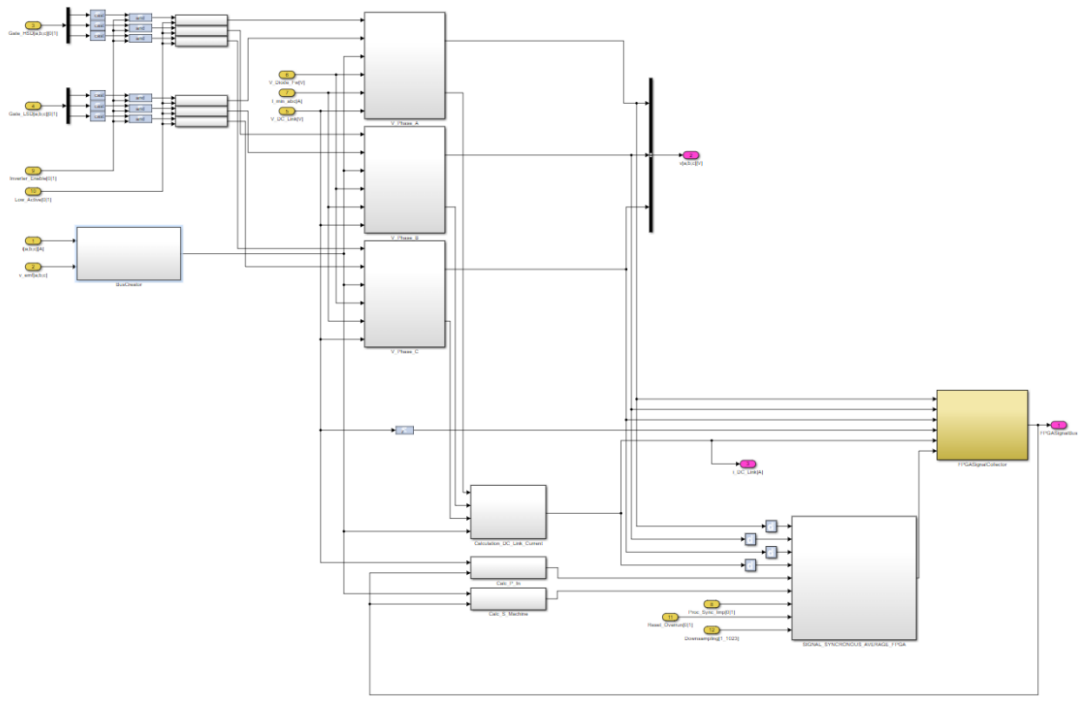


Figure 100 interior of the inverter dSPACE (screen from Simulink)

Regarding the motor the possible developments that could be made are basically two, that is the possibility to operate from generator and the possibility to operate in the field weakening zone. Both improvements can be taken from the characteristic curves of the engine.
The generator operation was not taken into account in the development of the thesis but is related to the fact that when the rotor is faster than the stator. This working condition is widely used in modern electric motors and is known as regenerative braking. The name comes from the fact that the motor operates as a generator because it generates electrical power. This is used to recharge the battery of the motor in order to ensure a' longer range and avoid wasting' energy. In more modern and sophisticated cars this possibility is leading to the development of the so-called "one-pedal driving", which allows vehicles to slow down only by stopping to press the accelerator pedal.

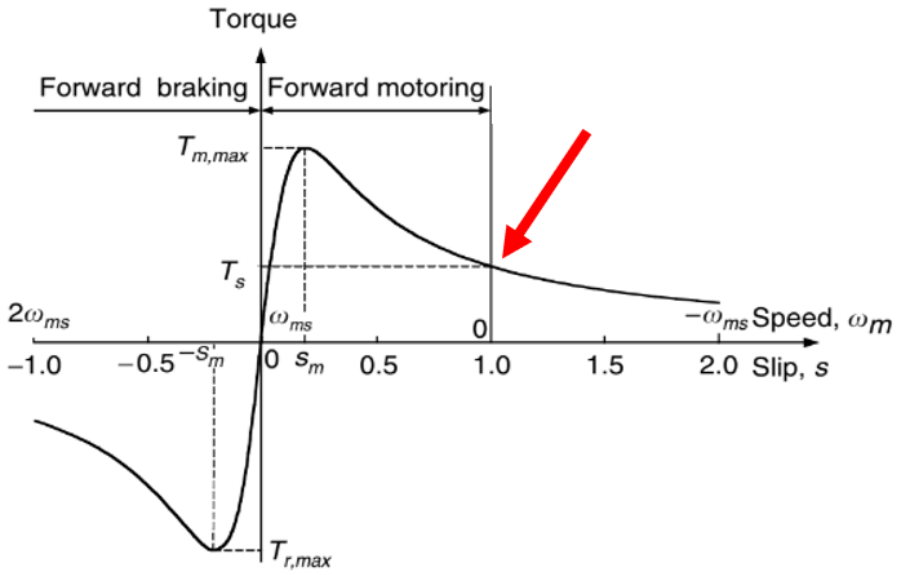


Figure 101 Regenerative braking (Modern Electric, Hybrid Electric, and Fuel Cell Vehicles. (Author: Mehrdad Ehsani, Yimin Gao))

The second improvement instead becomes very important in applications where it is required that the motor reaches very high rotation speeds, even in addition to the nominal speed. Field weakening is the possibility of bringing the engine to work at high speeds at the expense of a decrease in torque.
Below is a last image in which it is clear which area you are talking about.

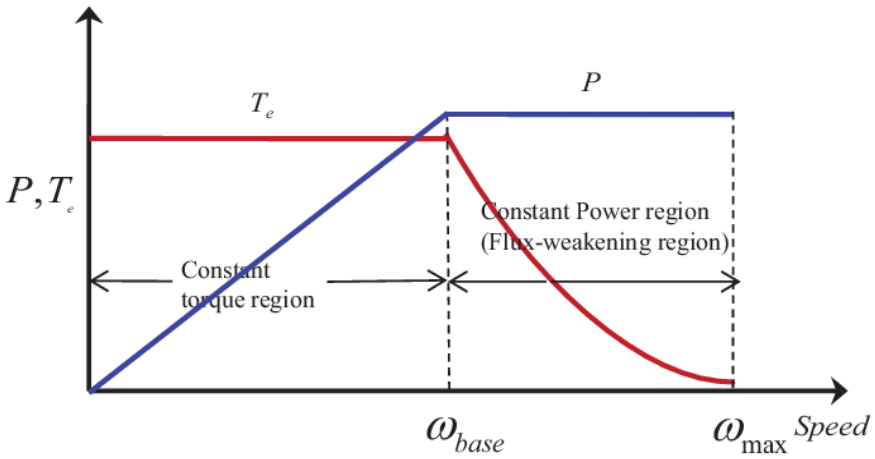


Figure 102 Field-weakening (semanticscholar.org)

THANKS

At the end of this long thesis journey that lasted months, it's time to write something more personal.

It seems only yesterday that for the first time I was presented the possibility of carrying out a project with those mysterious objects called HIL and FPGA. When I think about it again, I wonder why I chose to take a path full of topics never seen in years of university. The answer is simple and must be sought in the binomial Antonio Vitale-Kineton. A first and heartfelt thanks goes to them, to the reality that opened the doors and gave me the opportunity to start the thesis and the path of personal growth that led me to be where I am today. The reality in which I performed my internship has believed so much in me that, at the end of the project, I was also offered the opportunity to continue working with them, as a colleague and not as an "intern". I am extremely grateful for what has been given to me. A special thanks goes to Antonio, to whom I owe a lot: his example of professionalism and competence have been (and still are) a source of great inspiration and reflection. Related to the business I would like to thank many colleagues, who without having anything in return have helped a young student with commitment and passion. The list would be too long to list them one by one, but a special thanks goes to each of them.

Then I feel honored to be able to spend two words for my fellow thesis, Luca, Andreas and Fabio: I learned a lot from and with you. Your presence has gradually made the thesis work in a path of personal growth, all together. When I think about my thesis, in the near future, I will be very proud to say that I have coped with you. You were able to make a project that was so scary at the beginning enjoyable, and we became friends as colleagues. I wish everyone who has to start a journey to have the good fortune to find companions unique and special as you have been for me. Thank you, really.

Further on, I owe a huge thank you to Professor Alessandro Rizzo. I've been sniffing out the possibility of doing my thesis with you since the first year after my robotics exam. From the beginning it was clear that you had something more than the other practitioners for sympathy, professionalism and human approach. I'd ask you a thousand more times to do my thesis together.

Finally, a special thanks goes to all the people who were there, from my grandparents, the Nelli, my friends, Fulvione Nazionale, Andrea, Michela, Alessandro, Giuliano, Fabio, Alessia, Matteo, Simone, Benni, Gianluca, Erika, Joddan, Vito, Pierì, Gabriele. The list would still be very long, but in order not to become too long we stop here. All these people have always

been there for me, so much so that this is my thesis that concludes my path of study, but it is a goal of all, and that brings with it many people. Thanks for being there.

And finally, of course, my mom Mai'na gioia, my dad Ciccio, my sister Stuuupida, and my girlfriend PorGiGGiLek-Sube. To you I dedicate all the good that has happened in this journey. If today I am proud of what I have become it is only thanks to you. If in the future I should be able to make even half of what you have given me, I would be happy. You have been everything one needs. Simply, Thank you.

Let us all enjoy this goal together that new goals await us.

Ad meliora et maiora semper!

REFERENCES:

https://en.wikipedia.org/wiki/Electric_motor
https://guide.directindustry.com/it/come-scegliere-un-motore-elettrico/
https://www.landbelectric.com/download-document/82-basic-squirrel-cage-induction-motor-scim-overview.html
https://www.electrical4u.com/squirrel-cage-induction-motor/
https://www.biemmepiautoattrezzature.it/2018/10/30/i-motori-elettrici-trifase/#:~:text=Motore%20asincrono%20trifase%2C%20vantaggi%20e,sviluppo%20di%20calore%20nelle%20bobine.
https://www.chimica-online.it/fisica/motore-sincrono.htm
https://en.wikipedia.org/wiki/Motor_controller
https://www.ecutesting.it/categorie/panoramica-sulla-centralina-elettronica/
https://www.digikey.it/it/articles/field-oriented-control-of-small-dc-motors-put-drones-on-a-rising-flight-path
https://www.dday.it/redazione/34508/cose-linverter-la-tecnologia-che-ha-rivoluzionato-gli-elettrodomestici
https://it.mathworks.com/solutions/power-electronics-control/clarke-and-park-transforms.html
https://www.dspace.com/en/pub/home/applicationfields/foo/hil-testing.cfm
https://it.wikipedia.org/wiki/Software-in-the-loop
https://it.wikipedia.org/wiki/Hardware-in-the-loop
https://it.mathworks.com/solutions/power-electronics-control/hardware-in-the-loop.html
https://www.aptiv.com/en/insights/article/what-is-software-in-the-loop-testing